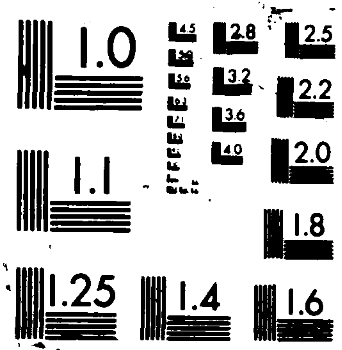
AD-A179 424 STAR SENSOR FINE ATTITUDE POINTING MODEL FOR A SPINNING 1/2
GEOSYNCHRONOUS SATELLITE(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. J V TAYLOR
UNCLASSIFIED DEC 86 AFIT/GA/AA/86D-14 F/G 22/1 NL



ROCOPY RESOLUTION TEST CHART

STAR SENSOR FINE ATTITUDE POINTING MODEL FOR A SPINNING GEOSYNCHRONOUS SATELLITE

THESIS

John V. Taylor IV
Captain, USAF

AFIT/GA/AA/86D-14

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

1

STAR SENSOR FINE ATTITUDE POINTING MODEL FOR A SPINNING GEOSYNCHRONOUS SATELLITE

THESIS

John V. Taylor IV
Captain, USAF

AFIT/GA/AA/86D-14

AFIT/GA/AA/86D-14

STAR SENSOR FINE ATTITUDE POINTING MODEL FOR A SPINNING GEOSYNCHRONOUS SATELLITE

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirement for the Degree of
Master of Science

by

John V. Taylor IV, M.S., B.S.
Capt USAF
Graduate Astronautical Engineering

December 1986

Approved for public release, distribution unlimited

Acknowledgements

I would like to express my sincere gratitude to the individuals who helped make this presentation possible. Foremost, I must thank my advisor, Dr. William Wiesel, whose keen academic knowledge guided me past any difficulties I encountered in conducting this research. I would also like to thank my thesis committee members, Capt's Ronald Hinrichsen and Stuart Kramer, and my typist/partner, Jo Ellen, for their assistance in eliminating errors from appearing in this document. Any mistakes which might remain herein are *not* theirs, but my own.

I thank the members of the GA-86D section for their help in making this report, as well as this entire academic endeavor, less difficult to accomplish.

Finally, I want to thank my daughters, Leigh and Crystal, for their tolerant endurance during this demanding period.

Jack Taylor

This document was prepared with the "FONTMASTER II" word processor package (xetec, inc.), using a Commodore 64 computer, and printed with a STAR SG-10 printer.

CONTENTS

List of Figures

List of Appendices

AFIT/GA/AA/86D-14

Abstract

A fine attitude determination model is developed for a spinning geosynchronous satellite, based upon stellar observations from a V-slit star scanner and ground supplied satellite ephemerides. A true attitude model is determined through numerical integration of Euler's moment equations for a torque-free, rigid, axisymmetric body, and kinematic relations which consist of pitch, roll, yaw, orientation angles. Next, the closed form solutions to the Euler equations are coupled with first-order approximate solutions of the kinematic relations to develop a second order kinematics model. Observation relations, relating stellar slit-plane crossing times, (a priori star identification assumed), to attitude states, are then developed. Finally, a nonlinear least-squares estimation algorithm is used to identify the full satellite attitude state. Simulation is tested for single scan and multiple-scan capability using exact and noise ridden data. Keywords: S[illegible] attitude [illegible] Star sensors; [illegible] h[illegible] [illegible]

STAR SENSOR FINE ATTITUDE POINTING MODEL
FOR A SPINNING GEOSYNCHRONOUS SATELLITE

Chapter One Introduction

Topic Description

Today's satellites utilize a variety of sensors and instruments for attitude determination. For a typical spinning satellite, data from a tachometer, sun sensor, earth sensor, and star sensor would be coupled with an ephemeris prediction model and fed through a ground-based computer's attitude prediction algorithm to yield the satellite's state (pitch,roll,yaw, and rates). In the quest for autonomous satellite navigation, simple and reliable models that accurately compute satellite attitude need to be developed. This report analyzes an attitude prediction model for a torque-free, spinning, rigid, axisymmetric, geosynchronous satellite based solely on satellite ephemerides, a star catalogue, and data obtained from a scanning star sensor. As onboard satellite computing capacity continues to increase, it is quite feasible that satellite-borne attitude determination schemes will soon replace ground based systems. For maximum reliability, these algorithms will require as much independence as possible. Optimal pointing accuracy will be achieved through a weighted reliance on all of the sensors, but in the event of satellite software or hardware anomolies, standalone capabilities would prove invaluable. An algorithm similar to the one presented in this paper might one day make up an independent portion of a satellite's attitude determination package.

Background

Satellite star sensors were conceived to assist in determining a satellite's attitude. Many of the early ideas for solution to the general problem of spacecraft attitude determination were probably first consolidated in open literature in the "Proceedings of the Symposium on Spacecraft Attitude Determination" (1:1-438). In that report, attitude determination based on stellar observations is a pervasive topic. The current need for efficient, independent computer programs for possible space-borne application gives the subject a renewed importance.

Model Description

Any attitude determination scheme based on stellar observations consists of an observer, a dynamics package, and an estimator. The estimator fits the data recorded by the observer with the motions described by the dynamics package and produces a best estimate of the attitude. In computer simulations, the functions of the observer must be augmented with a truth model, which tracks the attitude of the true state of the simulated satellite.

The observer can be a camera, a star tracker, or a star scanner. Star trackers operate by mechanically tracking a particular star. Data is recorded as the orientation between the tracker boresight axis and a satellite fixed reference frame. Star scanners, on the other hand, are generally fixed to the satellite, and they scan the heavens as the satellite moves. Data is recorded as the time when a star passes through the field-of-view of the scanner. The field-of-view usually contains one or more planar slits, called slit planes (2:562). For spinning satellites, the practical choice of star sensors is the scanner. Herein, a star scanner is modelled with the following specifications: +6.0 Magnitude Sensitivity; 3° Field-of-view; .205' Accuracy (3σ); 36°/sec Scan Rate. Comparing this with a first generation star sensor, the OSO-8 Star Scanner, with +4.0 Magnitude Sensitivity; 10° Field-of-view; 6' Accuracy (3σ); 30°/sec Scan Rate, (3: 224), it can be seen that the model specifications are more stringent, but not unfeasible. The scan rate mentioned above can be either a satellite characteristic, or, in the case where the scanner is motor-driven, a characteristic of the combined satellite dynamics and motor rate. A complete description of the star sensor model is given later in this report.

The attitude dynamics package describes the motion of the satellite. A dynamics package may be a piece of hardware or a mathematical model. An example of a combined hardware/software dynamics package would be an aircraft's inertial navigation system. For this report, a mathematical dynamics package was chosen to model. The dynamic equations consist of Euler's first order moment equations and kinematic equations which contain Euler orientation angles. Many papers written on this subject use Euler parameters instead of Euler orientation angles to describe the satellite's kinematics. While this choice avoids singularities which are inherent in the use of Euler orientation angles, it forces the use of an additional variable, and it removes physical clarity from the dynamic description. This report uses the familiar pitch, roll, and yaw orientation angles, and while it is acknowledged no solutions are possible for pitch angles of 90°, later assumptions prohibit this situation.

The assumptions of an axisymmetric rigid body in torque-free motion permit the solution of Euler's equations in closed form. It has been shown that making an axisymmetric assumption in near-axisymmetric situations leads to very small errors (2:568). The kinematic equations are next solved to first order by imposing small angle restrictions on the satellite's pitch, spin speed deviation, precession rate, precession angle, and orbital rate. For the problem of fine-attitude pointing on a nominally earth pointing satellite, these assumptions are not unreasonable. The first order kinematic solutions are then used to derive second order solutions, which make up the second half of the dynamics package. The general orbital dynamics problem is assumed to be solved independently for this simulation. A full description of the attitude dynamics routine is given in the next section of this report.

The final component of the attitude determination system is the estimator. Herein, a nonlinear least squares routine is chosen. It will be shown later that this routine is well suited for the problem at hand.

The truth model was simulated by numerical integrations of the exact dynamic equations which were fed through the observation relations.

Chapter Two Analytical Development

Overview

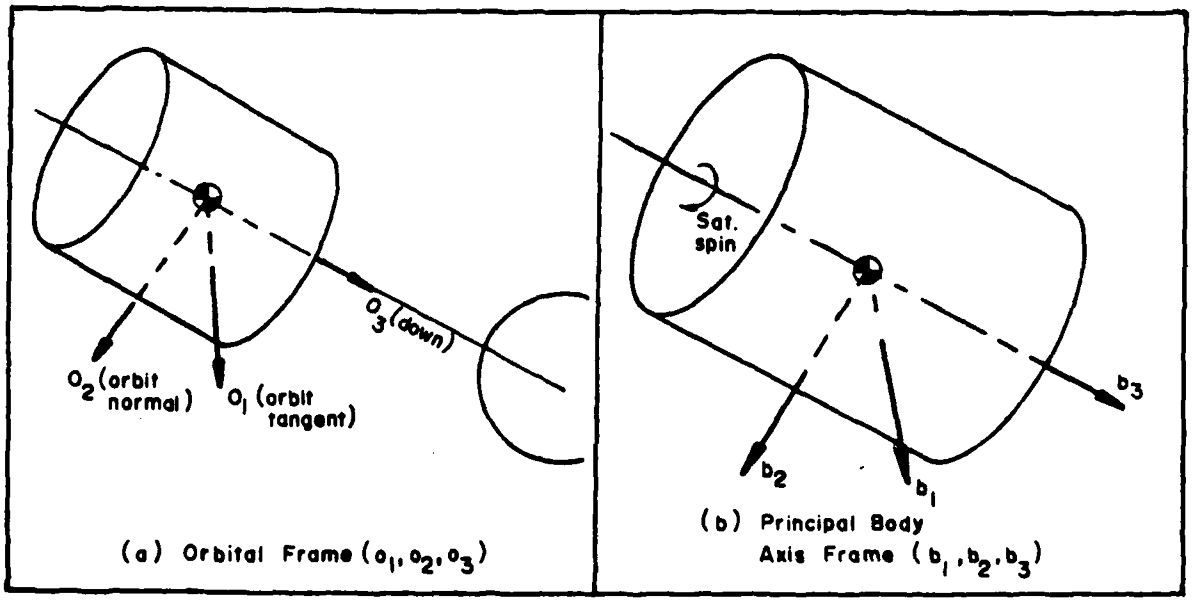
The computer model for attitude determination will contain, as stated earlier, four basic parts: the dynamics model, the observer model, the truth model, and the estimation algorithm.

The equations comprising the dynamics model are Euler's moment equations, which can be solved exactly for the torque-free rigid body, and a set of kinematic relations, whose solutions are derived through second order approximations. Next, the observation relations, equivalent to taking the inner product of the boresight axis of each slit plane of the star sensor with a star vector represented in the body frame, are derived. Then, the truth model, consisting of a bright star catalogue (4: H1-H31 & 5:Sec. VI) and a numerical integration of the exact state equations, is fully described. Finally, the nonlinear least squares estimation algorithm, which takes inputs from all other components to produce a state estimate, is explained.

Satellite Dynamics

Coordinate Systems

In order to define the satellite's dynamics, two coordinate systems are employed. The first reference frame, the orbital frame (o_1, o_2, o_3), centered at the satellite's mass center, consists of a pair of axes in the orbital plane (o_3 "down" from the satellite to the earth's center, and o_1 "eastward" along the orbital path), and an o_2 axis normal to the orbital plane (see Figure 1a). The second reference frame consists of a principal body axis set (b_1, b_2, b_3). b_3 is the symmetry axis of the satellite (nominally down-pointing), and b_1 and b_2 complete a right-handed set in the spin plane of the satellite (see Figure 1b).



(a) Orbital Frame (o_1, o_2, o_3)

(b) Principal Body Axis Frame (b_1, b_2, b_3)

Figure 1. Coordinate Reference Frames

Kinematics

The kinematic equations of motion are derived from a pitch (ψ_2), roll (ψ_1), yaw (ψ_3) rotation [2,1,3] of the orbital axes. Thus:

$$\begin{Bmatrix} b_1 \\ b_2 \\ b_3 \end{Bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{Bmatrix} o_1 \\ o_2 \\ o_3 \end{Bmatrix} \quad (1)$$

where:

$$a_{11} = \sin(\psi_1)\sin(\psi_2)\sin(\psi_3) + \cos(\psi_2)\cos(\psi_3) \quad (2a)$$

$$a_{12} = \cos(\psi_1)\sin(\psi_3) \quad (2b)$$

$$a_{13} = \sin(\psi_1)\cos(\psi_2)\sin(\psi_3) - \sin(\psi_2)\cos(\psi_3) \quad (2c)$$

$$a_{21} = \sin(\psi_1)\sin(\psi_2)\cos(\psi_3) - \cos(\psi_2)\sin(\psi_3) \quad (2d)$$

$$a_{22} = \cos(\psi_1)\cos(\psi_3) \quad (2e)$$

$$a_{23} = \sin(\psi_1)\cos(\psi_2)\cos(\psi_3) + \sin(\psi_2)\sin(\psi_3) \quad (2f)$$

$$a_{31} = \cos(\psi_1)\sin(\psi_2) \quad (2g)$$

$$a_{32} = -\sin(\psi_1) \quad (2h)$$

$$a_{33} = \cos(\psi_1)\cos(\psi_2) \quad (2i)$$

(6:423)

Use of this rotation scheme instead of the commonly used [3,1,3] Euler angle rotation moves the singularities inherent in these schemes out of the area of concern. (7:514)

The angular velocity of the body axes, with respect to the orbital axes, is:

$$\underline{\omega}_1{}^{b/o} = [\dot{\psi}_2\cos(\psi_1)\sin(\psi_3) + \dot{\psi}_1\cos(\psi_3)]\ \hat{b}_1 \qquad (3a)$$

$$\underline{\omega}_2{}^{b/o} = [\dot{\psi}_2\cos(\psi_1)\cos(\psi_3) - \dot{\psi}_1\sin(\psi_3)]\ \hat{b}_2 \qquad (3b)$$

$$\underline{\omega}_3{}^{b/o} = [-\dot{\psi}_2\sin(\psi_1) + \dot{\psi}_3\ \ \hat{b}_3] \qquad (3c)$$

(8:469)

Now, for a simplified orbital model, let the satellite be in an equatorial, circular, prograde orbit. Then, the angular velocity of the orbital axes, with respect to an inertial geocentric set, is:

$$\underline{\omega}^{o/I} = -\Omega\ \ \hat{o}_2 \qquad (4)$$

or, expressed in the body frame, using equation (1):

$$\underline{\omega}_1{}^{o/I} = -\Omega\cos(\psi_1)\sin(\psi_3)\ \ \hat{b}_1 \qquad (5a)$$

$$\underline{\omega}_2{}^{o/I} = -\Omega\cos(\psi_1)\cos(\psi_3)\ \ \hat{b}_2 \qquad (5b)$$

$$\underline{\omega}_3{}^{o/I} = \Omega\sin(\psi_1)\ \ \hat{b}_3 \qquad (5c)$$

Adding equations (3) and (5) yields the total angular velocity of the satellite, expressed in the body frame:

$$\omega_1 = \underline{\omega}_1^{b/I} = [-\Omega\cos(\psi_1)\sin(\psi_3) + \dot{\psi}_2\cos(\psi_1)\sin(\psi_3) + \dot{\psi}_1\cos(\psi_3)]\ \hat{b}_1 \qquad (6a)$$

$$\omega_2 = \underline{\omega}_2^{b/I} = [-\Omega\cos(\psi_1)\cos(\psi_3) + \dot{\psi}_2\cos(\psi_1)\cos(\psi_3) - \dot{\psi}_1\sin(\psi_3)]\ \hat{b}_2 \qquad (6b)$$

$$\omega_3 = \underline{\omega}_3^{b/I} = [\Omega\sin(\psi_1) - \dot{\psi}_2\sin(\psi_1) + \dot{\psi}_3]\ \hat{b}_3 \qquad (6c)$$

Equation (6) can be solved for $\dot{\psi}_1$, $\dot{\psi}_2$, and $\dot{\psi}_3$, to yield the kinematic equations of motion in first-order, nonlinear form:

$$\dot{\psi}_1 = \omega_1\cos(\psi_3) - \omega_2\sin(\psi_3) \qquad (7a)$$

$$\dot{\psi}_2 = \omega_1\sin(\psi_3)/\cos(\psi_1) + \omega_2\cos(\psi_3)/\cos(\psi_1) + \Omega \qquad (7b)$$

$$\dot{\psi}_3 = \omega_1\sin(\psi_3)\tan(\psi_1) + \omega_2\cos(\psi_3)\tan(\psi_1) + \omega_3 \qquad (7c)$$

(6:428)

Euler's Equations

In the principal body axis set, Euler's equations are:

$$A\dot{\omega}_1 + (C-B)\omega_2\omega_3 = M_1 \qquad (8a)$$

$$B\dot{\omega}_2 + (A-C)\omega_1\omega_3 = M_2 \qquad (8b)$$

$$C\dot{\omega}_3 + (B-A)\omega_1\omega_2 = M_3 \qquad (8c)$$

where A, B, and C are the principal moments of inertia, and M_1, M_2, and M_3 are the moment components.

Equation (8), when solved for $\dot{\omega}_1$, $\dot{\omega}_2$, $\dot{\omega}_3$, yields the attitude dynamic equations of motion in first order form:

$$\dot{\omega}_1 = ((B-C)/A)\omega_2\omega_3 + M_1/A \quad (9a)$$

$$\dot{\omega}_2 = ((A-C)/B)\omega_1\omega_3 + M_2/B \quad (9b)$$

$$\dot{\omega}_3 = ((A-B)/C)\omega_1\omega_2 + M_3/C \quad (9c)$$

For the case under study, it will be assumed that the satellite is symmetric about the b_3 axis. Thus: $A = B$.

Furthermore, if we let the inertia ratio $((A-C)/A)$ be represented by k, then Euler's equations reduce to:

$$\dot{\omega}_1 = k\omega_2\omega_3 + M_1/A \qquad (10a)$$

$$\dot{\omega}_2 = -k\omega_1\omega_3 + M_2/A \qquad (10b)$$

$$\dot{\omega}_3 = M_3/C \qquad (10c)$$

Torque-Free Motion

In the absence of external torques $(M_1=M_2=M_3=0)$, Euler's Equations can be solved in closed form:

$$\omega_1 = \omega_{10}\cos[\omega_{30}k(t-t_0)] + \omega_{20}\sin[\omega_{30}k(t-t_0)] \qquad (11a)$$

$$\omega_2 = -\omega_{10}\sin[\omega_{30}k(t-t_0)] + \omega_{20}\cos[\omega_{30}k(t-t_0)] \qquad (11b)$$

$$\omega_3 = \omega_{30} \qquad (11c)$$

where ω_{10}, ω_{20}, and ω_{30} are all constants of the motion.

The dynamic model to be used in this work will involve the torque-free solution just developed.

First Order Approximations of the Kinematic Equations

The solutions to Euler's equations (11) can be used to help integrate the kinematic equations (7), yielding the total dynamic description of the satellite's attitude in non-differential form. In order to complete the integrations, the kinematic equations will first be simplified. With a nominally "down-pointing" satellite, ψ_1 and ψ_2, as well as $\dot{\psi}_1$ and $\dot{\psi}_2$, will remain small. Also, with the o_3 and b_3 axes remaining nearly aligned, ω_1 and ω_2 will have values on the same order as ψ_1 and ψ_2 (i.e.: small). A final assumption will be that orbital rate (Ω) will be comparitively small. Using first-order small angle approximations for ψ_1 and ψ_2, equations (7) become:

$$\dot{\psi}_1 = \omega_1\cos(\psi_3) - \omega_2\sin(\psi_3) \tag{12a}$$

$$\dot{\psi}_2 = \omega_1\sin(\psi_3) + \omega_2\cos(\psi_3) + \Omega \tag{12b}$$

$$\dot{\psi}_3 = \omega_1\psi_1\sin(\psi_3) + \omega_2\psi_1\cos(\psi_3) + \omega_3 \tag{12c}$$

Neglecting second order terms in Ω, ψ_1, ψ_2, ω_1, and ω_2, equation (12c), to first-order, becomes:

$$\dot{\psi}_3 = \omega_3 \qquad (13)$$

Integrating:

$$\psi_3 = \psi_{30} + \omega_{30}(t-t_0) \qquad (14)$$

The results from equation (14), when inserted into the right hand sides of equations (12), will give first-order approximations for $\dot{\psi}_1$, $\dot{\psi}_2$, and $\dot{\psi}_3$:

$$\dot{\psi}_1 = \omega_1\cos[\psi_{30}+\omega_{30}(t-t_0)] - \omega_2\sin[\psi_{30}+\omega_{30}(t-t_0)] \qquad (15a)$$

$$\dot{\psi}_2 = \omega_1\sin[\psi_{30}+\omega_{30}(t-t_0)] + \omega_2\sin[\psi_{30}+\omega_{30}(t-t_0)] + \Omega \qquad (15b)$$

$$\dot{\psi}_3 = \omega_{30} \qquad (15c)$$

Substituting the solutions of Euler's equations (11) into equations (15) yields:

$$\begin{aligned}\dot{\psi}_1 &= \omega_{10}\cos[(k-1)\omega_{30}(t-t_0)-\psi_{30}] \\ &\quad + \omega_{20}\sin[(k-1)\omega_{30}(t-t_0)-\psi_{30}] \qquad (16a) \\ \dot{\psi}_2 &= -\omega_{10}\sin[(k-1)\omega_{30}(t-t_0)-\psi_{30}] \\ &\quad + \omega_{20}\cos[(k-1)\omega_{30}(t-t_0)-\psi_{30}] + \Omega \qquad (16b) \\ \dot{\psi}_3 &= \omega_{30} \qquad (16c)\end{aligned}$$

These equations can easily be integrated to yield:

$$\begin{aligned}\psi_1 = \psi_{10} &+ [\omega_{10}/((k-1)\omega_{30})]\sin[(k-1)\omega_{30}(t-t_0)-\psi_{30}] \\ &+ [\omega_{10}/((k-1)\omega_{30})]\sin(\psi_{30}) \\ &- [\omega_{20}/((k-1)\omega_{30})]\cos[(k-1)\omega_{30}(t-t_0)-\psi_{30}] \\ &+ [\omega_{20}/((k-1)\omega_{30})]\cos(\psi_{30})\end{aligned} \tag{17a}$$

$$\begin{aligned}\psi_2 = \psi_{20} &+ [\omega_{10}/((k-1)\omega_{30})]\cos[(k-1)\omega_{30}(t-t_0)-\psi_{30}] \\ &- [\omega_{10}/((k-1)\omega_{30})]\cos(\psi_{30}) \\ &+ [\omega_{20}/((k-1)\omega_{30})]\sin[(k-1)\omega_{30}(t-t_0)-\psi_{30}] \\ &+ [\omega_{20}/((k-1)\omega_{30})]\sin(\psi_{30}) + \Omega(t-t_0)\end{aligned} \tag{17b}$$

$$\psi_3 = \psi_{30} + \omega_{30}(t-t_0) \tag{17c}$$

Equations (17) give a first-order solution to the torque-free rigid body attitude dynamics problem.

Second Order Approximations of the Kinematic Equations

The order of the approximate solutions can be increased by substituting the first order approximations (eq. 17) into the right hand side of equations (12), and then integrating these new equations (ref. 9). When this is completed, the following relations are obtained:

$$\begin{aligned}
\psi_1=\psi_{10} &+ \omega_{10}d\{\sin(ct_1)+\sin(\psi_{30})\} - \omega_{20}d\{\cos(ct_1)-\cos(\psi_{30})\} \\
&+ [d^2(\omega_{10}{}^2+\omega_{20}{}^2)/2]\{\omega_{10}\cos(ct_1)+\omega_{20}\sin(ct_1)\}\ (t-t0) \\
&- \{dA_4\omega_{20} + [d^3(\omega_{10}{}^2+\omega_{20}{}^2)\omega_{10}/2]\}\ \{\sin(ct_1)+\sin(\psi_{30})\} \\
&- \{dA_4\omega_{10} + [d^3(\omega_{10}{}^2-\omega_{20}{}^2)\omega_{20}/2]\}\ \{\cos(ct_1)-\cos(\psi_{30})\} \\
&+ \{d^2A_6(\omega_{10}{}^2-\omega_{20}{}^2)/2\}\ \{\sin^2(ct_1)-\sin^2(\psi_{30})\} \\
&- \{d^2A_6\omega_{10}\omega_{20}/2\}\ \{\sin(2ct_1)+\sin(2\psi_{30})\} \\
&+ \{[(d\omega_{10})^3/6]-(d^3\omega_{10}{}^2\omega_{20}/2)\}\ \{\sin^3(ct_1)+\sin^3(\psi_{30})\} \\
&- \{d^3(\omega_{10}{}^2+\omega_{20}{}^2)\omega_{20}/6\}\ \{\cos^3(ct_1)-\cos^3(\psi_{30})\}
\end{aligned} \tag{18a}$$

$$\begin{aligned}
\psi_2=\psi_{20} &+ \omega_{10}d\{\cos(ct_1)-\cos(\psi_{30})\} + \omega_{20}d\{\sin(ct_1)+\sin(\psi_{30})\} \\
&+ \{\Omega-\ [d^2(\omega_{10}{}^2+\omega_{20}{}^2)/2][\omega_{10}\sin(ct_1)-\omega_{20}\cos(ct_1)]\}(t-t0) \\
&+ \{(A_6{}^2d\omega_{20}/2)+(dA_4\omega_{10})-(d^3\omega_{10}{}^2\omega_{20}/2)\} \\
&\ *\{\sin(ct_1)+\sin(\psi_{30})\} \\
&+ \{(A_6{}^2d\omega_{10}/2)-(dA_4\omega_{20})-(3d^3\omega_{20}{}^2\omega_{10}/2)\} \\
&\ *\{\cos(ct_1)-\cos(\psi_{30})\} \\
&+ \{2d^2A_6\omega_{10}\omega_{20}\}\ \{\sin^2(ct_1)-\sin^2(\psi_{30})\} \\
&+ \{(A_6d^2/2)(\omega_{10}{}^2-\omega_{20}{}^2)\}\ \{\sin(2ct_1)+\sin(2\psi_{30})\} \\
&+ \{d^3\omega_{10}{}^2\omega_{20}-[(d\omega_{20})^3/3]\}\ \{\sin^3(ct_1)+\sin^3(\psi_{30})\} \\
&+ \{d^3\omega_{10}\omega_{20}{}^2-[(d\omega_{10})^3/3]\}\ \{\cos^3(ct_1)-\cos^3(\psi_{30})\}
\end{aligned} \tag{18b}$$

$$\begin{aligned}
\psi_3=\psi_{30} &+ \omega_{30}(t-t0) - [d(\omega_{10}{}^2+\omega_{20}{}^2)/2](t-t0) \\
&+ (dA_6)\ \{\omega_{10}[\cos(ct_1)-\cos(\psi_{30})]+\omega_{20}[\sin(ct_1)+\sin(\psi_{30})]\} \\
&+ [d^2(\omega_{10}{}^2-\omega_{20}{}^2)/4]\ \{\sin(2ct_1)+\sin(2\psi_{30})\} \\
&+ [d^2\omega_{10}\omega_{20}]\ \{\sin^2(ct_1)-\sin^2(\psi_{30})\}
\end{aligned} \tag{18c}$$

where:

$$k = [(A-C)/A] \quad \text{(Inertia Ratio)} \tag{18a}$$

$$d = [(k-1)\omega_{30}]^{-1} \tag{18b}$$

$$ct_1 = (k-1)\omega_{30}(t-t0) - \psi_{30} \tag{18c}$$

$$A_6 = \psi_{10} + d\omega_{10}[\sin(\psi_{30})] + d\omega_{20}[\cos(\psi_{30})] \tag{18d}$$

$$A_4 = -[d^2\cos(\psi_{30})]\{[(\omega_{10}{}^2-\omega_{20}{}^2)/2]\sin(\psi_{30})+\omega_{10}\omega_{20}\cos(\psi_{30})\}$$
$$-d\psi_{10}[\omega_{10}\cos(\psi_{30})-\omega_{20}\sin(\psi_{30})] \tag{18e}$$

These second order kinematic approximations, coupled with the solutions to Euler's equations (eq. 7), complete the dynamic description of the satellite attitude model.

The solutions to the dynamic equations can be combined to form the attitude state vector {x}, consisting of state elements $\{\omega_1, \omega_2, \omega_3, \psi_1, \psi_2, \psi_3\}^T$. The matrix whose elements are made up of the partial derivatives of the state vector with respect to the state elements is called the [Φ] matrix. The [Φ] matrix normally plays a major role in an estimation algorithm. The [Φ] matrix is not used in the simulation developed in this research. In this estimation algorithm, the [T] matrix, which is the matrix of partial derivatives of the observation data vector with respect to the initial state vector, is derived numerically. A full discussion of this subject is included later in this report.

Observation Geometry

The inertial reference frame will be defined by the celestial sphere. Let the origin of the system be at the satellite's mass center. The X axis will point to the first point of Aries (γ), the Y axis will point east along the celestial equator, and the Z axis will point toward the celestial north pole. Since all data will consist of distant star sightings, parallax will be ignored (i.e.-the reference frame will be treated as though the origin were at the earth's center). Also, since the period of analysis will remain short, the motion effects of the earth's pole (precession, nutation, wobble) will be ignored. Star information will be stored as right ascension (α) and declination (δ). (see fig. 2).

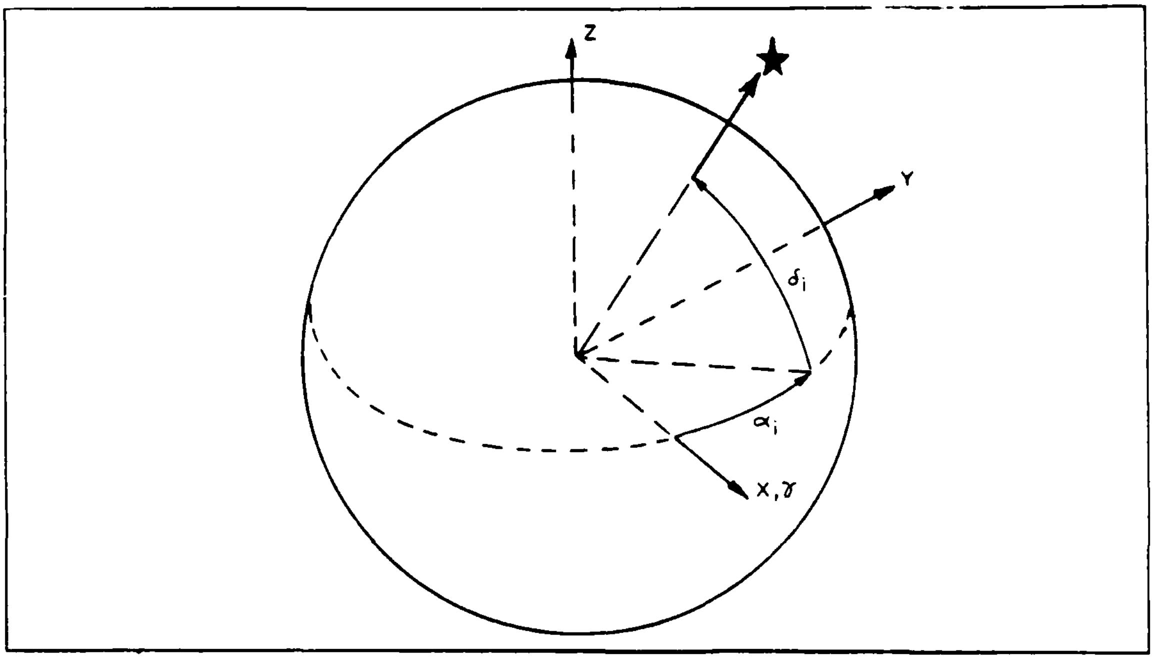


Figure 2. Stellar Geometry

Stellar data will be taken by a V-slit star sensor, recorded as first slit crossing time (t_1), and dwell time (t_d). A full description of the optical geometry will be shown later in this section.

Looking at the orientation of the orbital frame at a time (t_{go}) when the satellite crosses the +X axis, the following relationship is evident (see fig. 3):

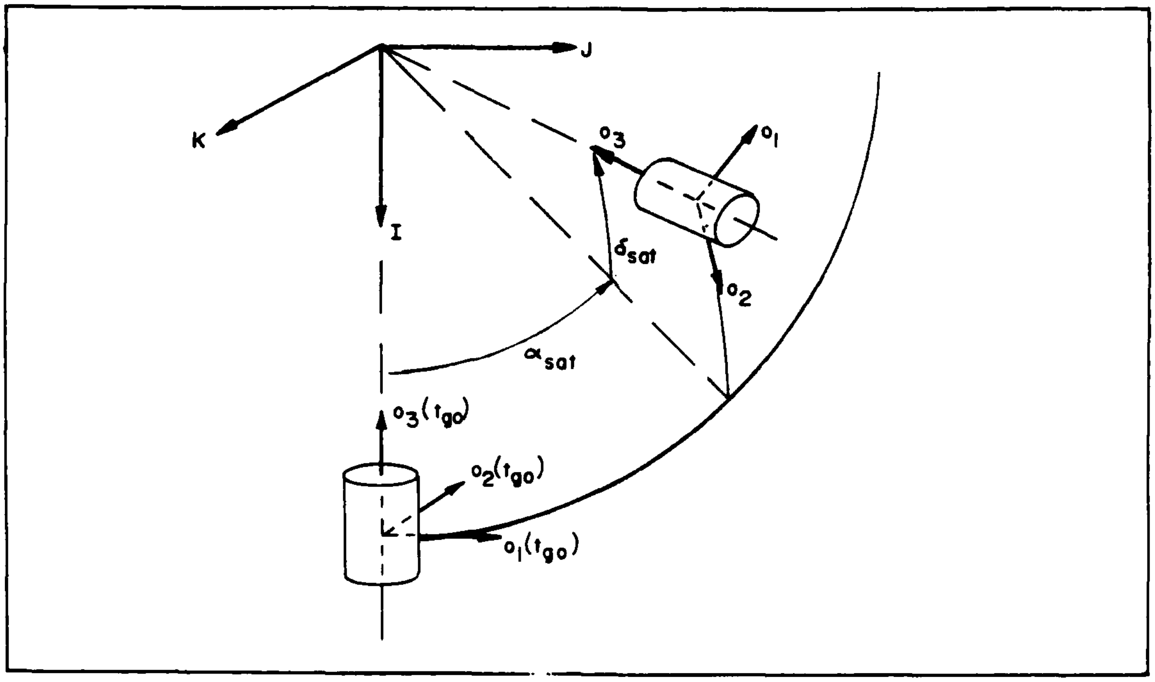


Figure 3. Orbital Reference Frame Orientation

$$\hat{o}_1(t_{go}) = \hat{J} \qquad (20a)$$

$$\hat{o}_2(t_{go}) = -\hat{K} \qquad (20b)$$

$$\hat{o}_3(t_{go}) = -\hat{I} \qquad (20c)$$

This relationship assumes, for the time being, that the orbital plane is not inclined with respect to the celestial equator. In matrix notation, equation (20) becomes:

$$\begin{Bmatrix} o_1(t_{go}) \\ o_2(t_{go}) \\ o_3(t_{go}) \end{Bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -1 \\ -1 & 0 & 0 \end{bmatrix} \begin{Bmatrix} I \\ J \\ K \end{Bmatrix} \qquad (21)$$

Now, at some arbitrary time t, given the satellite's right ascension and declination (α_s, δ_s), the relationship between the orbital frame and the inertial frame (see fig. 3) becomes:

$$\begin{Bmatrix} o_1(t) \\ o_2(t) \\ o_3(t) \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\delta_s) & -\sin(\delta_s) \\ 0 & \sin(\delta_s) & \cos(\delta_s) \end{bmatrix} \begin{bmatrix} \cos(\alpha_s) & 0 & \sin(\alpha_s) \\ 0 & 1 & 0 \\ -\sin(\alpha_s) & 0 & \cos(\alpha_s) \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -1 \\ -1 & 0 & 0 \end{bmatrix} \begin{Bmatrix} I \\ J \\ K \end{Bmatrix} \qquad (22)$$

Multiplying the second and third matrices yields:

$$\begin{Bmatrix} o_1(t) \\ o_2(t) \\ o_3(t) \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\delta_s) & -\sin(\delta_s) \\ 0 & \sin(\delta_s) & \cos(\delta_s) \end{bmatrix} \begin{bmatrix} -\sin(\alpha_s) & \cos(\alpha_s) & 0 \\ 0 & 0 & -1 \\ -\cos(\alpha_s) & -\sin(\alpha_s) & 0 \end{bmatrix} \begin{Bmatrix} I \\ J \\ K \end{Bmatrix} \qquad (23)$$

For the purpose of this paper, the orbital model will consist of a geocentric satellite in two-body motion. In this case, $\delta_s = 0$, and, if we reference all time measurements to a crossing of the first point of Aries (t_{go}), then we can solve for the satellite's right ascension as:

$$\alpha_s = \Omega(t-t_{go}) \quad (24)$$

It is noteworthy that the attitude prediction to be developed herein could easily handle any degree of complexity orbital model, so long as the satellite's ephemerides were known quantities, and the satellite's orbit produced no short-term perturbations which might alter the orientation dynamics model. In the simplified case we have developed:

$$\begin{Bmatrix} o_1 \\ o_2 \\ o_3 \end{Bmatrix} = \begin{bmatrix} -\sin[\Omega(t-t_{go})] & \cos[\Omega(t-t_{go})] & 0 \\ 0 & 0 & -1 \\ -\cos[\Omega(t-t_{go})] & -\sin[\Omega(t-t_{go})] & 0 \end{bmatrix} \begin{Bmatrix} I \\ J \\ K \end{Bmatrix} \quad (25)$$

Rearranging, the inertial coordinates, in terms of the orbital frame, become:

$$\begin{Bmatrix} I \\ J \\ K \end{Bmatrix} = \begin{bmatrix} -\sin[\Omega(t-t_{go})] & 0 & -\cos[\Omega(t-t_{go})] \\ \cos[\Omega[t-t_{go})] & 0 & -\sin[\Omega(t-t_{go})] \\ 0 & -1 & 0 \end{bmatrix} \begin{Bmatrix} o_1 \\ o_2 \\ o_3 \end{Bmatrix} \qquad (26)$$

Now, given a known star position (α_i, δ_i), (see fig. 4) the coordinates can be transformed into the I J K system's as:

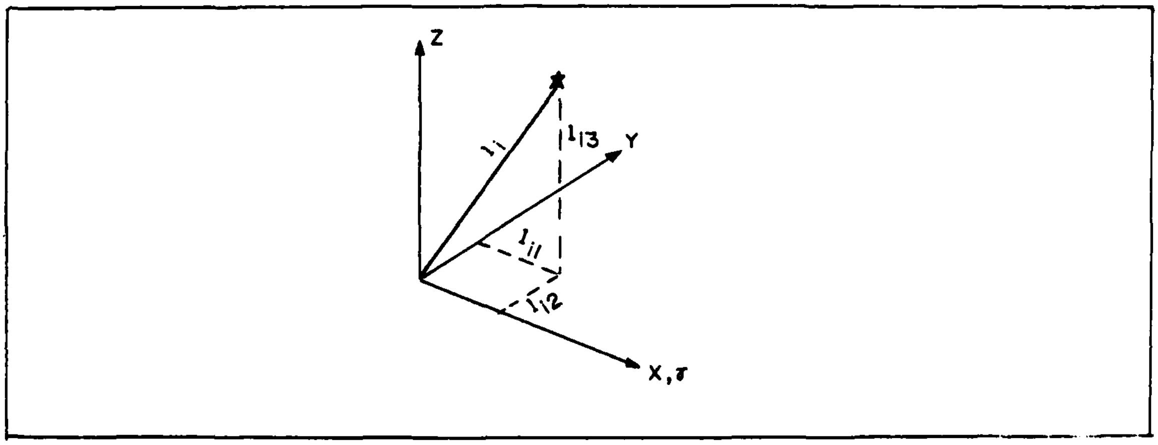


Figure 4. Star Vector Definition

$$\hat{l}_i = \cos(\alpha_i)\cos(\delta_i)\ \hat{I} + \sin(\alpha_i)\cos(\delta_i)\ \hat{J} + \sin(\delta_i)\ \hat{K} \quad (27)$$

Now, combining equations (26) and (27) with the relation between the orbital axes and the body axes yields:

$$\begin{Bmatrix} l_1 \\ l_2 \\ l_3 \end{Bmatrix} = \begin{Bmatrix} \cos(\delta_i)\cos(\alpha_i) \\ \cos(\delta_i)\sin(\alpha_i) \\ \sin(\delta_i) \end{Bmatrix}^T \begin{bmatrix} -\sin[\Omega(t-t_{go})] & 0 & -\cos[\Omega(t-t_{go})] \\ \cos[\Omega(t-t_{go})] & 0 & -\sin[\Omega(t-t_{go})] \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} R \end{bmatrix} \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \end{Bmatrix} \qquad (28)$$

where:

[R]=

$$\begin{bmatrix} \cos(\psi_2) & 0 & \sin(\psi_2) \\ 0 & 1 & 0 \\ -\sin(\psi_2) & 0 & \cos(\psi_2) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\psi_1) & -\sin(\psi_1) \\ 0 & \sin(\psi_1) & \cos(\psi_1) \end{bmatrix} \begin{bmatrix} \cos(\psi_3) & -\sin(\psi_3) & 0 \\ \sin(\psi_3) & \cos(\psi_3) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (29)$$

Expanding the [R] matrix will make it more convenient for use later. The expansion will be:

$$[R] = [R1(\psi_2)][R2(\psi_1)][R3(\psi_3)] \qquad (30)$$

where:

$$[R1(\psi_2)] = \begin{bmatrix} \cos(\psi_2) & 0 & \sin(\psi_2) \\ 0 & 1 & 0 \\ -\sin(\psi_2) & 0 & \cos(\psi_2) \end{bmatrix} \qquad (31)$$

$$[R2(\psi_1)] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\psi_1) & -\sin(\psi_1) \\ 0 & \sin(\psi_1) & \cos(\psi_1) \end{bmatrix} \qquad (32)$$

$$[R3(\psi_3)] = \begin{bmatrix} \cos(\psi_3) & -\sin(\psi_3) & 0 \\ \sin(\psi_3) & \cos(\psi_3) & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (33)$$

Star Sensor Operation

This paper will utilize a v-slit star sensor to obtain stellar data. Although a major portion of the operation of a stellar attitude determination scheme involves star identification, the method to be developed herein will assume a priori star identification with each star sighting. The star sensor will be defined to have the first slit aligned parallel with the b_3 axis, with the b_1 axis out the boresight axis, and the b_2 axis will complete the dextral set (see Figure 5). Physical misorientations could be easily accounted for (ref 7:218-221), but they will be neglected in this report.

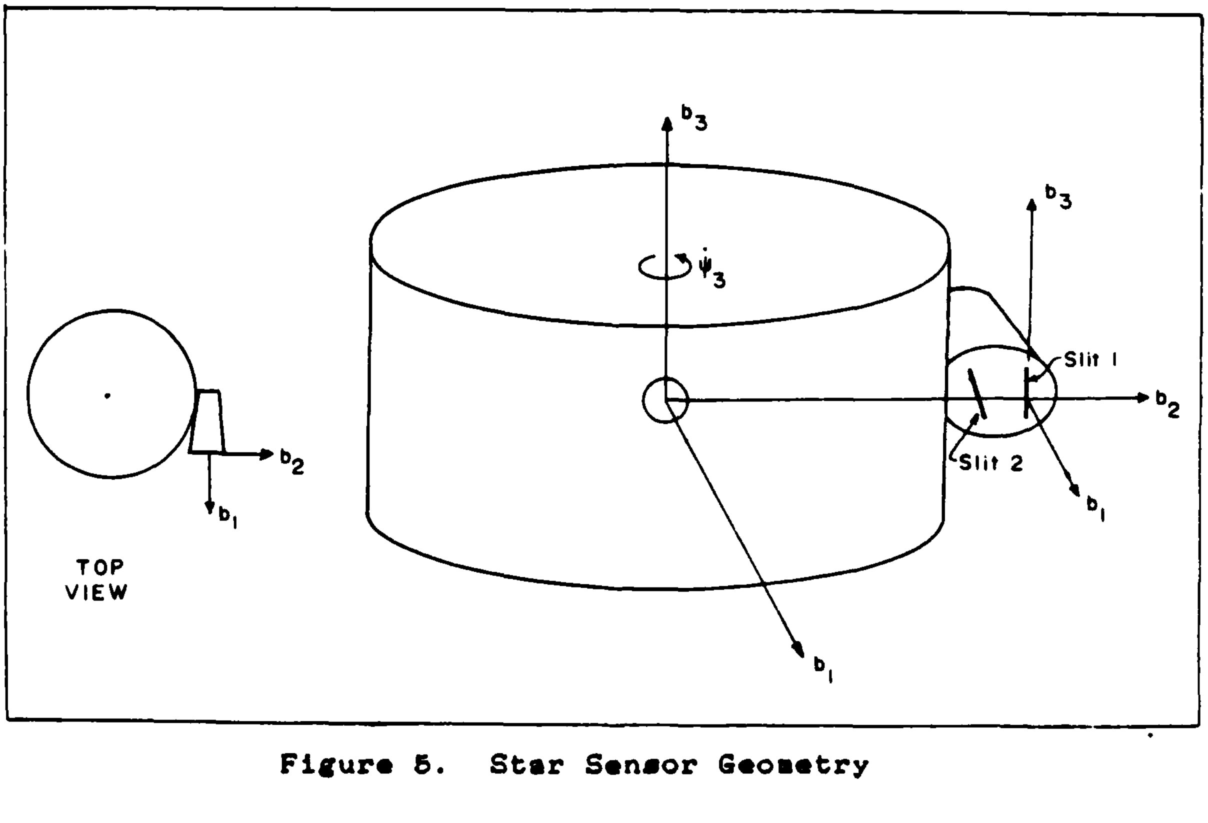


Figure 5. Star Sensor Geometry

Each slit of the star sensor has a planar field of view. The sensor is fixed to the satellite, and thus scans the sky as the satellite spins. The width of the field of view will be taken as three degrees.

At a time (t) when a star enters the field of view of slit one and is identified, the following relations exist (see Figure 6):

$$l_i(t) \cdot b_2 = 0 \quad (\text{"no" error}) \tag{34a}$$

$$l_i(t) \cdot b_1 = \cos(\delta_{1i}(t)) \tag{34b}$$

$$l_i(t) \cdot b_3 = \sin(\delta_{1i}(t)) \tag{34c}$$

$$\delta_{1i}(t) \leq 1.5 \text{ degrees} \tag{34d}$$

After a short time period (t_δ), the star will normally pass through slit two, and a new geometric relationship exists (see Figure 6).

This new relationship can be described in terms of a new body axis set (c_1, c_2, c_3) as:

$$l_i(t+t_\delta) \cdot c_2 = 0 \quad (\text{"no" error}) \tag{35a}$$

$$l_i(t+t_\delta) \cdot c_1 = \cos(\delta_{2i}(t+t_\delta)) \tag{35b}$$

$$l_i(t+t_\delta) \cdot c_3 = \sin(\delta_{2i}(t+t_\delta)) \tag{35c}$$

$$\delta_{2i}(t+t_\delta) \leq 1.5/\cos(\theta_i) \text{ degrees} \tag{35d}$$

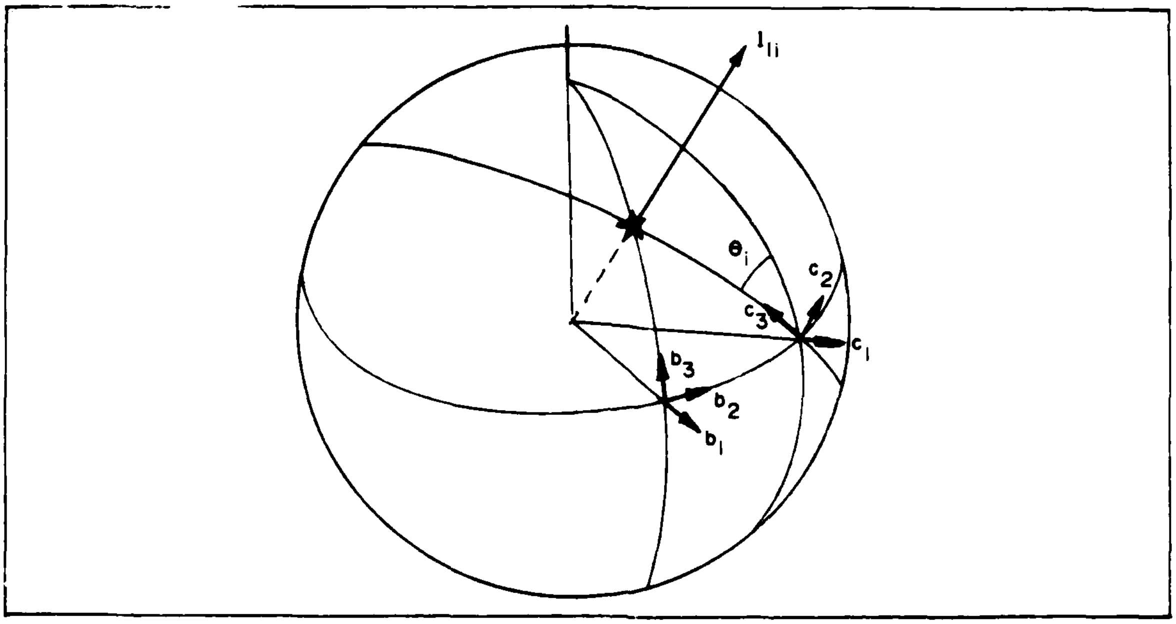


Figure 6. Star Slit Reference Frames

The body c_1, c_2, c_3 frame is linked to the principal body frame through two angles, θ_z and θ_i, which are properties of the star sensor's design (see Figure 7). From figure 7, the choice of the v-slits in the star sensor becomes apparent: the v-slits allow the declination of the star in the body frame to be determined. The relationship between c_1, c_2, c_3 and b_1, b_2, b_3 is:

$$\begin{Bmatrix} c_1 \\ c_2 \\ c_3 \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_i) & \sin(\theta_i) \\ 0 & -\sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \begin{bmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 \\ \sin(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \end{Bmatrix} \quad (36)$$

which reduces to:

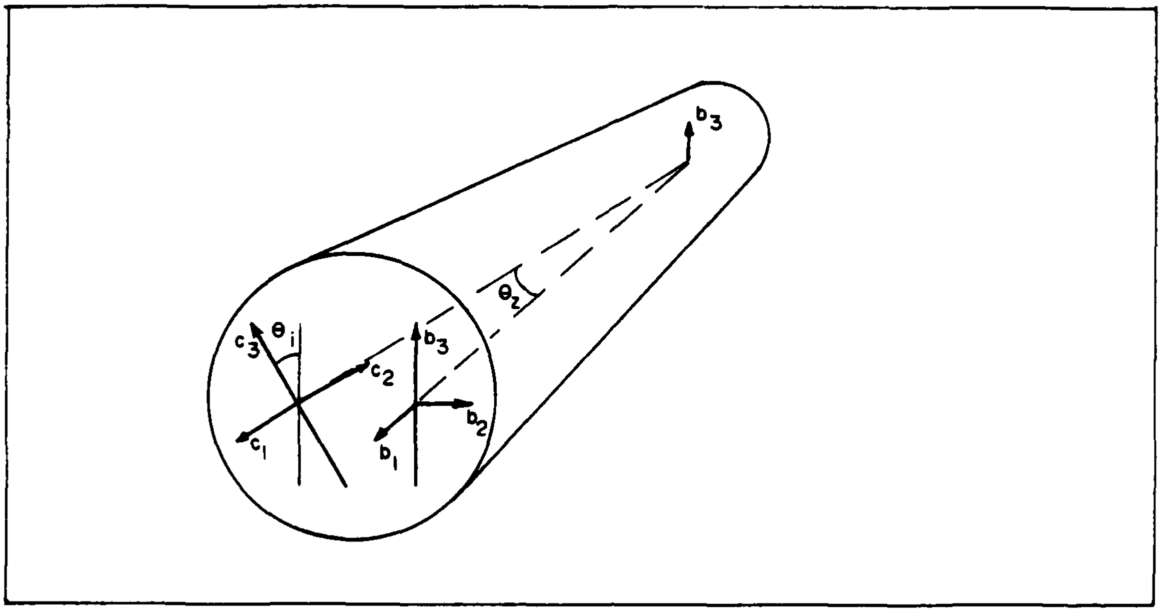


Figure 7. Star Scanner Design Angles

$$\begin{Bmatrix} c_1 \\ c_2 \\ c_3 \end{Bmatrix} = \begin{bmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 \\ \cos(\theta_i)\sin(\theta_z) & \cos(\theta_i)\cos(\theta_z) & \sin(\theta_i) \\ -\sin(\theta_i)\sin(\theta_z) & -\sin(\theta_i)\cos(\theta_z) & \cos(\theta_i) \end{bmatrix} \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \end{Bmatrix} \qquad (37)$$

Referring to equation (28), letting the star position vector and orbit position matrix be represented by $P(\alpha_i, \delta_i)$ and $R\Omega(t-t_{go})$, respectively, then combining this expression with equations (34),(35), and (37), yields:

$$\{P(\alpha_i,\delta_i)\}^T[R\Omega(t-t_{go})][R1(\psi_2)][R2(\psi_1)][R3(\psi_3)]\ \{b\} = \{P_{b1i}\} \qquad (38)$$

$$\{P(\alpha_i,\delta_i)\}^T[R\Omega(t-t_{go})][R1(\psi_2)][R2(\psi_1)][R3(\psi_3)][R\theta] = \{P_{b2i}\} \qquad (39)$$

where:

$$\left\{P(\alpha_i,\delta_i)\right\}=\begin{Bmatrix}\cos(\alpha_i)\cos(\delta_i)\\ \sin(\alpha_i)\cos(\delta_i)\\ \sin(\delta_i)\end{Bmatrix} \tag{40}$$

$$\left[R\Omega(t-t_{go})\right]=\begin{bmatrix}-\sin[\Omega(t-t_{go})] & 0 & -\cos[\Omega(t-t_{go})]\\ \cos[\Omega(t-t_{go})] & 0 & -\sin[\Omega(t-t_{go})]\\ 0 & -1 & 0\end{bmatrix} \tag{41}$$

$$\left[R\theta\right]=\begin{bmatrix}\cos(\theta_z) & -\sin(\theta_z) & 0\\ \cos(\theta_i)\sin(\theta_z) & \cos(\theta_i)\cos(\theta_z) & \sin(\theta_i)\\ -\sin(\theta_i)\sin(\theta_z) & -\sin(\theta_i)\cos(\theta_z) & \cos(\theta_i)\end{bmatrix}\begin{Bmatrix}b_1\\ b_2\\ b3\end{Bmatrix} \tag{42}$$

$$\left\{P_{b1i}\right\}=\begin{Bmatrix}\cos(\delta_{1i}(t))\\ 0\\ \sin(\delta_{1i}(t))\end{Bmatrix} \tag{43}$$

$$\left\{P_{b2i}\right\}=\begin{Bmatrix}\cos(\delta_{2i}(t+t_\delta))\\ 0\\ \sin(\delta_{2i}(t+t_\delta))\end{Bmatrix} \tag{44}$$

As a final simplification, let the left hand side of equations (38) and (39) be represented by Obs1 and Obs2, respectively. Then, using only the equality portions of these equations, the observation relations reduce to:

$$Obs1_2 = 0 \qquad (45a)$$

$$Obs2_2 = 0 \qquad (45b)$$

Truth Model

An operating star scanner would obtain inputs from stellar slit transits, which, after being time-tagged by the satellite's reference clock, would be sent to the attitide determination package. These slit crossing times would be directly dependent on the satellite's dynamic characteristics. In computer simulations, the observation inputs must be provided by the truth model. The model used in this effort, as stated earlier, assumes a satellite in geosynchronous, two body orbit, with nominally earth-pointing attitude. Under these assumptions, the reference attitude, (the orbital reference frame), is established by the time elapsed since t_{go}, and this reference rotates about the o_2 axis at constant rate (Ω) of $7.292115856\text{x}10^{-5}$ rad/sec (10:429). Thus, given the true initial attitude state parameters (ω_{10}, ω_{20}, ω_{30}, Ψ_{10}, Ψ_{20}, Ψ_{30}) and the initial time, with respect to t_{go}, the true state of the satellite can be tracked by numerically integrating the exact first order differential equations. Knowledge of the true state permits the star vector to be transformed into the star sensor slit frame, using the transformations developed in the last section. Watching the $b_2(c_2)$ component of a star vector for a sign change allows determination of a slit plane crossing time. After checking field-of-view constraints, a star sighting can be recorded.

For this research, a star data base, which is accessed by the truth model algorithm, was compiled from the Bright Stars, J1984.5 table of "The Astronautical Almanac" (4:H1-H31). This data base lists a circumpolar band of stars, approximately 30° wide, centered on a nominal satellite position of 0° right ascension at $t_{go}=0$ (see figure 8). The list contains 301 stars of visual magnitude +6.0 or brighter (see Table 1, Appendix A).

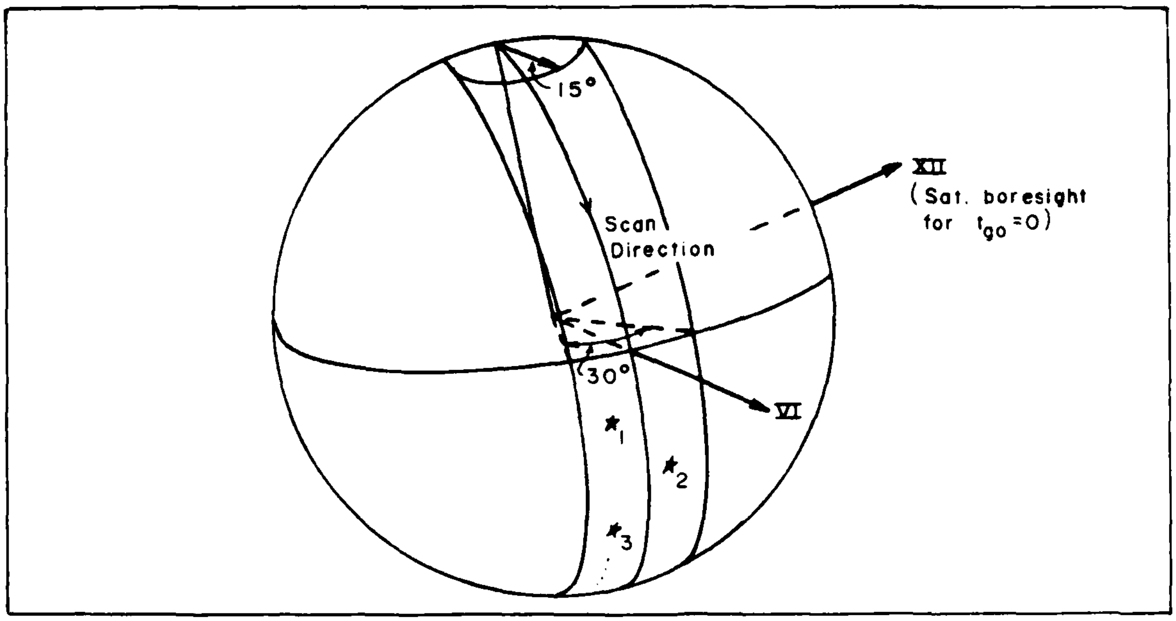


Figure 8. Truth Model Star Field

Estimator

The final component of the attitude determination package is the estimator. Here, the truth data is combined with the approximated dynamics and observation relations to form the state estimate. Due to the nonlinear nature of the dynamics and observations relations, some nonlinear estimation algorithm is desired. The choice made for this research is a nonlinear least squares algorithm.

Nonlinear Least Squares Algorithm

A typical nonlinear least squares algorithm operates in the following manner:

I) Propagate the state vector to the observation time. Also, obtain the state transition matrix $\Phi(t,t_0)$.

II) For each observation, calculate:

a) $$r_i = z_i - G(x_{ref}(t_i),t_i) \tag{46}$$

b) $$H_i = (\partial G/\partial x)\ [(x_{ref}(t_i),t_i)]_{xref} \tag{47}$$

c) $$\Phi(t_i,t_0) = \partial x(t_i)/\partial x(t_0) \tag{48}$$

d) $$T_i = H_i\Phi_i(t_i,t_0) \tag{49}$$

III) Add new terms to the running sums of the matrix:

$$\Sigma\ T_i{}^T Q_i{}^{-1} T_i$$

and the vector:

$$\Sigma\ T_i{}^T Q_i{}^{-1} r_i$$

IV) Compute the covariance of the correction:

$$P_{\delta x} = [\Sigma\ T_i{}^T Q_i{}^{-1} T_i]^{-1} \qquad (50)$$

and the state correction at epoch:

$$\delta x(t_0) = P_{\delta x}\ \Sigma\ T_i{}^T Q_i{}^{-1} r_i \qquad (51)$$

V) Correct the reference attitude state:

$$x_{ref}(new) = x_{ref}(old) + \delta x(t_0) \qquad (52)$$

VI) Repeat steps I through V until convergence is achieved. (Check residuals for valid convergence.)

(11:68 & 12:30,31)

The components needed to employ this algorithm will now be described.

Observation Vector (G)

The observation relations developed earlier (eq. 45), when expressed as a vector, make up the observation vector {G(x$_i$,t$_i$}, which can be expressed as:

$$\{G(x_i,t_i)\} = \begin{Bmatrix} Obs1(x_{i1},t_{i1}) \\ Obs2(x_{i2},t_{i2}) \end{Bmatrix} \qquad (53)$$

Some simplification can be obtained by treating the information from each star scanner slit as a separate data point. In this manner, the data vector {z} is reduced to a scalar value, and the observation vector {G} relation is reduced to two scalar relations. Which relation is used is determined by the slit making the observation.

Gradient of the Observation Vector [H]

Now, having tranformed the observation relation into a scalar relation, the gradient of the observation relation with respect to the state vector, the [H] matrix, becomes a row vector relation. In order to obtain [H], the following partial derivatives will be required:

$$\partial/\partial\psi_2\ [R1(\psi_2)] = \begin{bmatrix} -\sin(\psi_2) & 0 & \cos(\psi_2) \\ 0 & 0 & 0 \\ -\cos(\psi_2) & 0 & -\sin(\psi_2) \end{bmatrix} \qquad (54)$$

$$\partial/\partial\psi_1\ [R2(\psi_1)] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -\sin(\psi_1) & -\cos(\psi_1) \\ 0 & \cos(\psi_1) & -\sin(\psi_1) \end{bmatrix} \quad (55)$$

$$\partial/\partial\psi_3\ [R3(\psi_3)] = \begin{bmatrix} -\sin(\psi_3) & -\cos(\psi_3) & 0 \\ \cos(\psi_3) & -\sin(\psi_3) & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (56)$$

With respect to the state variables, all other partial derivatives are zero, Assigning matrix nomenclature to the above equations, let:

$$[HR1(\psi_2)] = \partial/\partial\psi_2\ [R1(\psi_2)] \quad (57)$$

$$[HR2(\psi_1)] = \partial/\partial\psi_1\ [R2(\psi_1)] \quad (58)$$

$$[HR3(\psi_3)] = \partial/\partial\psi_3\ [R3(\psi_3)] \quad (59)$$

Now [H] can be assembled as:

$$[H] = [\ h_1 \quad h_2 \quad h_3 \quad h_4 \quad h_5 \quad h_6\ ] \quad (60)$$

The elements of [H] are:

$$h_1 = \partial G/\partial\omega_1 \quad (61a)$$

$$h_2 = \partial G/\partial\omega_2 \quad (61b)$$

$$h_3 = \partial G/\partial\omega_3 \quad (61c)$$

$$h_4 = \partial G/\partial \psi_1 \qquad (61d)$$

$$h_5 = \partial G/\partial \psi_2 \qquad (61e)$$

$$h_5 = \partial G/\partial \psi_3 \qquad (61f)$$

Since the observation relations are dependent on the observation slit, the elements of [H] will also vary with the observation slit. For slit 1 star sightings, the elements of [H] are:

$$h_1 = 0 \qquad (62a)$$

$$h_2 = 0 \qquad (62b)$$

$$h_3 = 0 \qquad (62c)$$

$$h_4 = \mathrm{El}_2 \text{ of } \{\{P(\alpha_i,\delta_i)\}^T[R\Omega(t-t_{go})][R1(\psi_2)]$$
$$* [HR2(\psi_1)][R3(\psi_3)]\}^T \qquad (62d)$$

$$h_5 = \mathrm{El}_2 \text{ of } \{\{P(\alpha_i,\delta_i)\}^T[R\Omega(t-t_{go})][HR1(\psi_2)]$$
$$* [R2(\psi_1)][R3(\psi_3)]\}^T \qquad (62e)$$

$$h_6 = \mathrm{El}_2 \text{ of } \{\{P(\alpha_i,\delta_i)\}^T[R\Omega(t-t_{go})][R1(\psi_2)]$$
$$* [R2(\psi_1)][HR3(\psi_3)]\}^T \qquad (62f)$$

For slit 2 star sightings, the elements of [H] are:

$$h_1 = 0 \qquad (63a)$$

$$h_2 = 0 \qquad (63b)$$

$$h_3 = 0 \qquad (63c)$$

$$h_4 = \mathrm{El}_2 \text{ of } \{\{P(\alpha_i,\delta_i)\}^T[R\Omega(t-t_{go})][HR1(\psi_2)]$$

$$* \ [R2(\psi_1)][R3(\psi_3)][R(\theta_i,\theta_z)]\}^T \qquad (63d)$$

$$h_5 = El_2 \text{ of } \{\{P(\alpha_i,\delta_i)\}^T[R\Omega(t-t_{go})][R1(\psi_2)]$$

$$* \ [HR2(\psi_1)][R3(\psi_3)][R(\theta_i,\theta_z)]\}^T \qquad (63e)$$

$$h_6 = El_2 \text{ of } \{\{P(\alpha_i,\delta_i)\}^T[R\Omega(t-t_{go})][R1(\psi_2)]$$

$$* \ [R2(\psi_1)][HR3(\psi_3)][R(\theta_i,\theta_z)]\}^T \qquad (63f)$$

From equations (60-63), it can be seen that the elements of [H] for slit 1 observations are determined from element 2 of a vector, which, if that vector is post-multiplied by the [Rθ] matrix, yields the elements of [H] for slit 2 observations. Problems which might arise from the small difference in observation times of a particular star by each star sensor slit are thus avoided by treating each sighting as separate data. All that is now required to form the residual is the transformation of the observation relation into a relation which estimates a slit crossing time.

Computing the residual {r}

The truth model gives the time for a given slit crossing. This parameter, t_{act}, permits solution of the state estimate. In order to compute the residual ($t_{act}-t_{est}$), it is necessary to determine the estimate of the slit crossing time from the estimated state. Since t is an implicit variable in the dynamics and in the observation relation, some iterative scheme will be required to find t_{est}. The observation relations lend themselves readily to a Newton-Rhaphson iteration technique, and this scheme will now be developed. The form of each observation relation is:

$$g_i(x,t) = 0 \quad (i=1,2) \qquad (64)$$

The Newton-Rhaphson method to solve for t will be:

$$t_{new} = t_{old} - g_i/(dg_i/dt) \qquad (65)$$

The actual slit crossing time will be used as the initial seed for each iteration. In order to find the time derivatives of each observation relation, the chain rule will be applied, as follows:

$$dg_i(x,t)/dt = \partial g_i/\partial t + (\partial g_i/\partial x)(\partial x/\partial t) \quad (66)$$

The only new expressions needed here are the partial derivatives of the observation relations with respect to time. Recalling the observation relations (eqs. 38,39) the only components of these equations which are explicit in t are the elements of:

$$[R\Omega(t-t_{go})]$$

The partial derivative of this expression with respect to t is:

$$\partial/\partial t[R\Omega(t-t_{go})] = -\Omega\begin{bmatrix}\cos[\Omega(t-t_{go})] & 0 & -\sin[\Omega(t-t_{go})] \\ \sin[\Omega(t-t_{go})] & 0 & \cos[\Omega(t-t_{go})] \\ 0 & 0 & 0\end{bmatrix} \quad (67)$$

Now, the first terms on the right-hand side of equation (66) become:

$$\dot{g}_1 = El_2 \text{ of } \{(P(\alpha_i,\delta_i))\}^T[\partial/\partial t[R\Omega(t-t_{go})]][R1(\psi_2)]$$
$$* [R2(\psi_1)][R3(\psi_3)]]\} \quad (68)$$

$$\dot{g}_2 = El_2 \text{ of } \{(P(\alpha_i,\delta_i))\}^T[\partial/\partial t[R\Omega(t-t_{go})]][R1(\psi_2)]$$
$$* [R2(\psi_1)][R3(\psi_3)][R(\theta_i,\theta_z)]\} \quad (69)$$

The other two terms required are $(\partial g/\partial x)$ and (dx/dt), which are already developed as [H] and the first order state differential equations. Now, equation (66) becomes:

$$dg/dt_i = \dot{g}_i + [H_i]\ \{\partial x_i/\partial t_i\} \qquad (70)$$

where i(=1,2) signifies the applicable star sensor slit.

Substitution of equation (70) into equation (66) yields the basic equation of the Newton-Rhaphson scheme, which, after iterating to a specified tolerance, gives an estimation of the slit crossing time. The residual scalar, r_i, can be calculated as:

$$r_i = t_{act i} - t_{est i} \qquad (71)$$

Data Gradient [T]

Having found the residuals, the gradient of the data scalar with respect to the initial state, [T], is the only remaining relation required for the assembly of the estimate correction covariance. The derivation of [T] is as follows:

$$r_i = (t_{act i} - t_{est i}) = [H_i] \{\delta x_i(t_i)\} \qquad (72)$$

where:

$$\{\delta x_i(t_i)\} = [\Phi(t_i, t_0)] \{\delta x(t_0)\} \qquad (73)$$

Substitution of equation (73) into equation (72) yields:

$$r_i = [H_i] [\Phi(t_i, t_0)] \{\delta x(t_0)\} \qquad (74)$$

The [T] matrix, (herein a row vector), is simply a shorthand notation for the product of [H] and [Φ]:

$$[T_i] = [H_i] \, [\Phi(t_i, t_0)] \qquad (75)$$

In terms of the data scalar and the state, the [T] matrix can be considered to be:

$$[T_i] = \partial t_i / \partial x_0 \qquad (76)$$

The problem of t being an implicit variable in the observation relations now resurfaces. This problem is circumvented by obtaining the [T] matrix numerically using a finite differencing technique. For each observation time t_i, an estimated slit crossing time is computed. By varying each initial estimated state variable by a small amount, a new estimated slit crossing time can be obtained for each variable change. The approximate relationship which results is:

$$[T_i] \approx \Delta t_i / \Delta x_0 \qquad (77)$$

Use of the approximate relationship for [T] eliminates the need to compute the [Φ] matrix.

Data Covariance Matrix [Q]

The data covariance matrix is a property of the satellite hardware. It is a numerical declaration of the precision of the instrument's ability to record the data. A simplified approach is taken in this problem by treating the data covariance as a scalar value. Thus, the value of q will simply become the square of the standard deviation of the observation time for a particular star:

$$q_i = (\sigma_{t_i})^2 \tag{78}$$

Now, all components required to compute the covariance of the estimate correction have been derived.

Covariance of the estimate correction [P]

The final component of the least squares estimation algorithm to be computed is the covariance of the estimate correction, or, as it is commonly called, the [P] matrix. As shown by equation (50), the [P] matrix is obtained by inverting the matrix described by:

$$[T_i{}^T Q_i{}^{-1} T_i]$$

With the modifications described above, the resulting expression for the [P] matrix in this simulation is:

$$[P] = q[\Sigma T_i{}^T T_i]^{-1} \qquad (79)$$

Modified Nonlinear Least Squares Algorithm

By incorporating all modifications into the typical nonlinear least squares algorithm described earlier, the following estimation algorithm emerges:

I) From the truth model, obtain the observation times. (Add noise if desired).

II) Propagate the state vector estimate to the observation time.

III) For each observation, obtain:

a) $g_i(x_{est_i}), t_i)$

b) $[H_i] = (\partial g/\partial x)$

c) $\partial g/\partial t$

d) dx/dt

IV) From components of step III, compute t_{est}

V) Repeat steps III and IV until a converged t_{est} is obtained. Form residual:

$$r_i = t_{act_i} - t_{est_i}$$

VI) Vary each initial state estimate variable, and repeat steps III-V to obtain:

$$T_i \approx \Delta t_i / \Delta x_0$$

VII) Add new terms to the running sums of the matrix:

$$q^{-1}[\Sigma T_i^T T_i]$$

and the vector:

$$q^{-1}[\Sigma T_i^T r_i]$$

VIII) Compute the covariance of the correction:

$$P_{\delta x} = q[[\Sigma T_i^T T_i]^{-1}]$$

and the state correction at epoch:

$$\delta x(t_0) = q[P_{\delta x}]\{\Sigma T_i^T r_i\}$$

IX) Correct the estimated attitude state:

$$x_{est}(new) = x_{est}(old) + \delta x(t_0)$$

X) Repeat steps I through IX until convergence is achieved. (Check residuals for valid convergence.)

The remainder of this research will be devoted to the development of a computer simulation which incorporates the algorithms which have been described in this section. Program implementation and simulation results will be presented.

Chapter Three Program Implementation

A major portion of the effort in this thesis involves the creation of a working simulation of a satellite attitude state prediction model. This chapter is devoted to taking a brief qualitative look at the programs used in the simulation. Also discussed are some of the hardwired constraints used in the program, and the reasoning behind the constraint choices (where pertinent). For a complete program listing, see Appendix B.

Program Overview

The program used to simulate the satellite attitude estimation consists of a main controller program, 16 subroutines, and two input files. All program code is written in FORTRAN 77 language, and one IMSL subroutine (LEQT2F) (ref 13:LEQT2F), as well as one IMSL function (GGNQF) (ref 13:GGNQF), are utilized.

Input Files

There are two input files for the estimation program, a Bright Star Catalog, "stardat" (App B25-B30), and a initial conditions input file, "guessin" (App B14). The Bright Star Catalog contains 301 stars, listing each star's right ascension and declination (in radians), as well as the star's identification number and visual magnitude. The list was developed from Table 1 (App. A1-A8) and put into usable format by a utility program "starcnv.f" (see appendix C1-C2). For a visualization of how the starfield might appear, consider an earth-pointing satellite positioned on the equator at the Greenwich meridian. If the satellite is spinning clockwise (viewed from behind the satellite looking towards earth), with the star scanner slit 1 boresight axis pointing along the earth's inertial Y axis (east), then an observer looking through the star scanner would see a star field as depicted in Figure 9 over a one scan period.

The initial conditions input file contains: the initial attitude state vector (in rads/sec;rads), the initial attitude state vector estimate (same units), initial sidereal time (seconds from J1984.5 first point of Aries), start time with respect to initial sidereal time (seconds), the number of epoch updates to be processed, and the number of scans to be propagated between epoch updates. Other parameters included in this input file are: the truth model numerical integration timestep (typically 0.1 sec), the number of ordinary differential equations to be integrated (always 6), the maximum number of iterations to be allowed by the estimator (10), a noise flag which determines if noise is to be added to the truth model data,

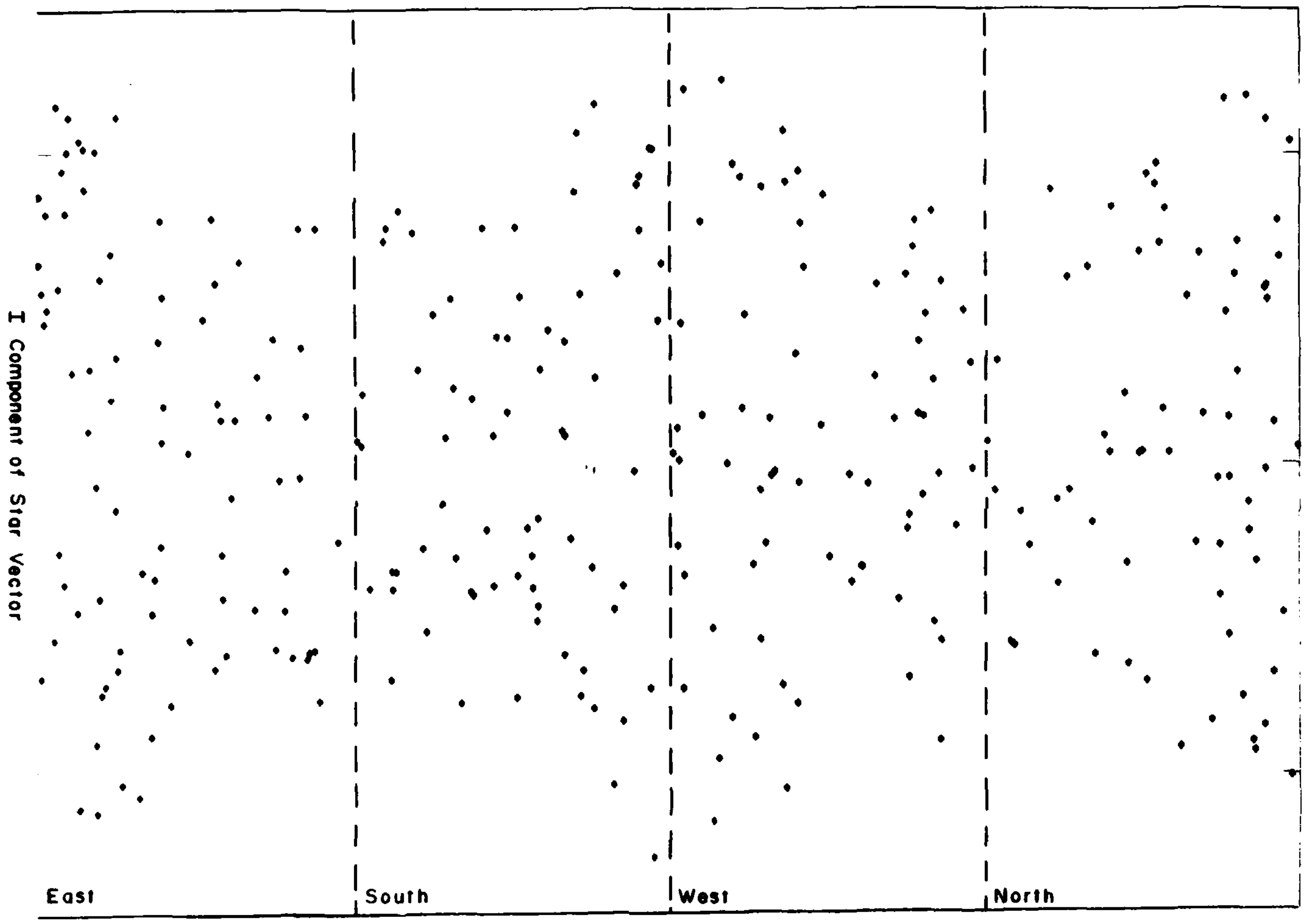


Figure 9. Star Field for One Scan Period

and a noise seed for the IMSL function's pseudo-random normal number generator.

Main Program Operation

The controlling program for the attitude estimation package, "estimator.f" (see app B2-B5), reads the initial conditions input file, controls the overall program flow, and produces the desired output information in a usable format. After reading the initial conditions input file, the initial conditions are stored in array format, and are printed to the output file. For each epoch update, the following processing occurs:

1) Input data for an estimation sequence are initialized.

2) Program control is transferred to subroutine "truth" (App B35-B37), where the truth model star sighting times for one scan period are determined from true initial conditions, stellar data, observation relations, and propagation of the true state. Star sighting information is stored in a temporary data file "sitings" (App D1-D2), and program control is passed back to the main program.

3) Program control is passed to subroutine "torder" (App B32-B33), which reads the "sitings" file, adds

noise to the truth model data (if required), prints noise information to the output file, time-orders the star sighting information, and stores the adjusted sighting information in temporary data file "stars" (App D3-D4). Program control is then passed back to the main program.

4) Next, program control is passed to subroutine "tresid" (App B34), where a Newton-Rhaphson iteration procedure is used to compute the estimated star observation times. True star sighting times, observation relations, approximate dynamics, and first-order exact state differential equations are used to obtain results. True observation times, estimated observation times, and stellar information are stored in temporary data file "statfile" (App D5-D7). Program control is then passed back to the main routine.

5) Program control is then passed to subroutine "dtdx0" (App B10), where a finite differencing technique is used to compute the [T] row vector for each star sighting. The procedures explained in the previous step are used to obtain results. Star information, true and estimated slit crossing times, and [T] data are stored in temporary data file "dtfile" (App D8-D9), and control is passed back to the main program.

6) Finally, program control is passed to subroutine "coverian" (App B7,B8), which processes the residuals to obtain the [P] matrix and the estimated state correction vector $\{x_0\}$. In this routine, the residuals and $\{x_0\}$ are printed to the output file. This routine also determines whether or not a converged state estimate has been achieved. If the state estimate $\{x_0\}$ has converged, the [P] matrix is printed to the output file. Program control, along with an estimate convergence flag, are then passed back to the main program.

7) Steps 1 through 6 are repeated until either a coverged state estimate is achieved, or the program fails when the maximum number of iteration cycles is exceeded.

After completing one estimation cycle and obtaining a converged state estimate at epoch, the main program next propagates the true and estimated states forward in time to the next epoch time. States are periodically printed to the output file. At each state printing time (set to one typical scan period), subroutine "comperr" (App B6), is called. This subroutine computes the vector magnitude of the total pointing error and prints this error, with the time of occurrence, to temporary data file "errlist" (App D10-D11).

Finally, after the program has completed all estimation propagation cycles, subroutine "ploterr" (App B24) reads the error data file "errlist", puts this information into plottable format, and stores the information in data file "errplot" (D14-D17).

Other Routines

Before discussing the hardwired parameters used in this simulation, the subroutines which have not already been specifically discussed will be presented.

Subroutine "theslrhs" (App B31) contains the exact first-order differential state equations. This routine is accessed by the numerical integration subroutine, and by the subroutine which processes the Newton-Rhaphson iteration for the estimated star observation time.

Subroutine "haming" (App B15-B17) is a fourth-order predictor-corrector numerical integration routine. It is used to propagate the true state equations forward in time. This algorithm was developed by Professor Anderson, Harvard University, in the 1960's. The version used in this simulation was provided by Dr. William Wiesel, Professor of Astronautical Sciences, AFIT.

Subroutine "dyno2" (App B12,B13) contains the second order approximate solution to the state attitude dynamics. Given an initial time, state, and time of concern, this routine computes the approximate state at the time of concern. A similar program "dynol" (App B11), contains the first order approximate solutions to the state attitude dynamics. With very minor program modifications, the estimation routine can be run with first order approximate dynamics in place of the current second order model.

Subroutine "observ2" (App B20-B23) contains the matrix relations required to compute the observation relations (g_i), the [H] row vector, or the partial derivative of the observation relation with respect to time ($gdot_i$). The calling routine provides star information, time, sidereal time, state, and a specification flag. The specification flag signals this routine which relation is required. "observ2" utilizes two matrix operation routines: "matxmat" (App B18), which multiplies two matrices, and "vecxmat" (App B40), which premultiplies a matrix by a row vector. The matrix manipulation routines were provided by fellow AFIT student Capt Keith Greer, GAE/86D.

Subroutine "tstate" (App B38,B39) drives the Newton-Rhaphson iteration scheme which is used to compute the [T] row vector.

Subroutine "noise" (App B19) uses IMSL function ggnqf to provide a one-sigma pseudo random normal time error. This time error is added to the true slit-crossing time to generate "noisy" truth data where required.

Hardwired Parameters

Several parameter constants are embedded within the estimator programs, but could be altered with minor program editing. These alterable constants will now be discussed.

The size of the input star data base is fixed at 301 stars spanning almost a 30^o wide longitudinal swath of the sky. Increasing the size of this list would require edition of the STAR.DB table, but the list could be easily diminished by editing either the main program or "starcnv. f" to discriminate according to visual magnitude.

The estimation package works with an underlying assumption of a nominal satellite spin rate of 5 revolutions per minute. A single estimation loop processes all data observed in a typical single scan period (12 seconds). In order to reduce the processing time involved in generating the truth model data, the truth subroutine was programmed only to look for star sightings for a time period equal to current epoch time + 12 seconds. Also, after a star observation is recorded, no second sighting for that star is permitted. Altering the typical spin rate of the satellite would require changing the parameter "tend" in the main program to a value approximately equal to the typical scan period (in seconds). Altering the truth model operation further would require more extensive program changes, and this topic will be discussed later in this report.

Star scanner physical parameters θ_i and θ_z are currently set to 15° and 0.5°, respectively, and they reside in subroutines "observ2" and "truth". The characteristics of the star scanner could be altered by simply editing these parameters. The scanner's field of view, also hardwired, is set to three degrees in subroutine "truth". Changing the parameter "fov" in "truth" is all that is required to alter the scanner's field of view.

The orbital rate parameter, Ω, carries with it many implications. In essence, this parameter summarizes the assumptions of a satellite reference attitude model which arises from a circular, two-body orbit having 0° inclination. It also might appear to imply that the reference attitude maintains its nominal earth-pointing state without torques. This is not the case, however, for although the governing state equations assume torque-free motion, the estimation algorithm could easily handle the presence of small, impulsive, attitude correction torques (so long as the true state corrections were presented to the truth model). Analysis of a more complex equatorial reference attitude model would require replacing the Ω parameter in routines "theslrhs" and "observ2" with a subroutine which described the reference attitude rotation rate. The study of inclined orbits would require a revised derivation of the observation relations.

Residual rejection criteria are established in subroutine "covarian". Currently, these parameters are set to 3 seconds for the first two estimation iterations, and 3 sigma for succesive iterations. If all residuals are rejected on one pass, the constraints are widened by an order of magnitude. For different case studies, it might be desirable to edit this rejection scheme.

The finite differencing technique used in this effort uses a forward-differencing approach, with the initial state being varied by a finite difference amount of 0.1%. Problems might arise for cases where initial conditions approach zero. Different criteria could easily be arranged by small editions to the "covarian" subroutine.

For this research, the inertia ratio ((A-C)/A) is set to a value of -1. This represents a typical, stable, tuna-can type of satellite with a spin moment of inertia equal in magnitude to twice the magnitude of the transverse moments of inertia. To study a different case, the parameter "ak" in the "thelrhs" and "dyno" routines would have to be edited.

In the subroutine "truth", a simplification is incorporated which locks sightings made by slit-2 of the star scanner to slit 1 observations. Essentially the field-of-view of slit 2 is adjusted so that all stars observed by slit 1 are also observed by slit 2. For a case study involving large satellite precession rates, this criterion might need to be adjusted.

The final hardwired constraints used in this simulation are the tolerances in the numerical integration routine "haming", and the Newton-Rhaphson convergence (subroutine "tstate"). While these tolerances proved adequate for this case study, it might be desirable to verify these parameters in a different analysis.

Now that a full description of the program operation and theoretical assumptions have been presented, results from several case studies can be presented.

Chapter Four <u>Results</u>

This section includes results from 8 case studies, as well as results obtained while verifying some of the subroutine algorithms. During this presentation, qualitative, as well as quantitative, results are compiled. The sequence used to present the results parallels the sequence utilized to develop the simulation program.

Typical Input Files

For the case studies which will follow, two typical cases were considered. For the first case, all true initial conditions were set to zero except for the spin rate (5 revolutions per minute), and the initial spin angle (0.8 radians). This case was chosen to depict a precession-free situation which should produce predictable results. For the second case, it was desired to find a general initial true state which contained arbitrary (small) deviations from the first case. To achieve this goal, the following true initial conditions were chosen:

$X_{10} = 0.01$	rad/sec	(0.57296°/sec)
$X_{20} = 0.05$	rad/sec	(2.86479°/sec)
$X_{30} = 0.5223598776$	rad/sec	(29.9290°/sec)
$X_{40} = 0.05$	rad	(2.86479°)
$X_{50} = 0.05$	rad	(2.86479°)
$X_{60} = 0.8$	rad	(45.8366°)

Estimated initial conditions were chosen to be close or equal to true initial conditions.

Dynamics Checkout Package

The first component of the attitude estimation package to be verified was the dynamics algorithm. The results for a single scan run with case two true initial conditions show that, qualitatively, both the first order and second order approximate dynamic models closely matched the numerically integrated exact equations. Graphs of the ψ_1, ψ_2, and ψ_3 states are shown in Figures 10, 11 and 12. (Results from ω states are not shown since these equations are solved exactly in closed form).

While the single-scan dynamics plots showed close agreement between approximate and truth model dynamics solutions, they did not clearly show the second order approximate equations presented any advantage over the first order approximate relations. For that reason, the Φ relations were next plotted in a similar fashion for identical initial conditions. Examples of these results, shown in Figures 13, 14 and 15, clearly show the superiority of the second order approximate dynamics over the first order approximations.

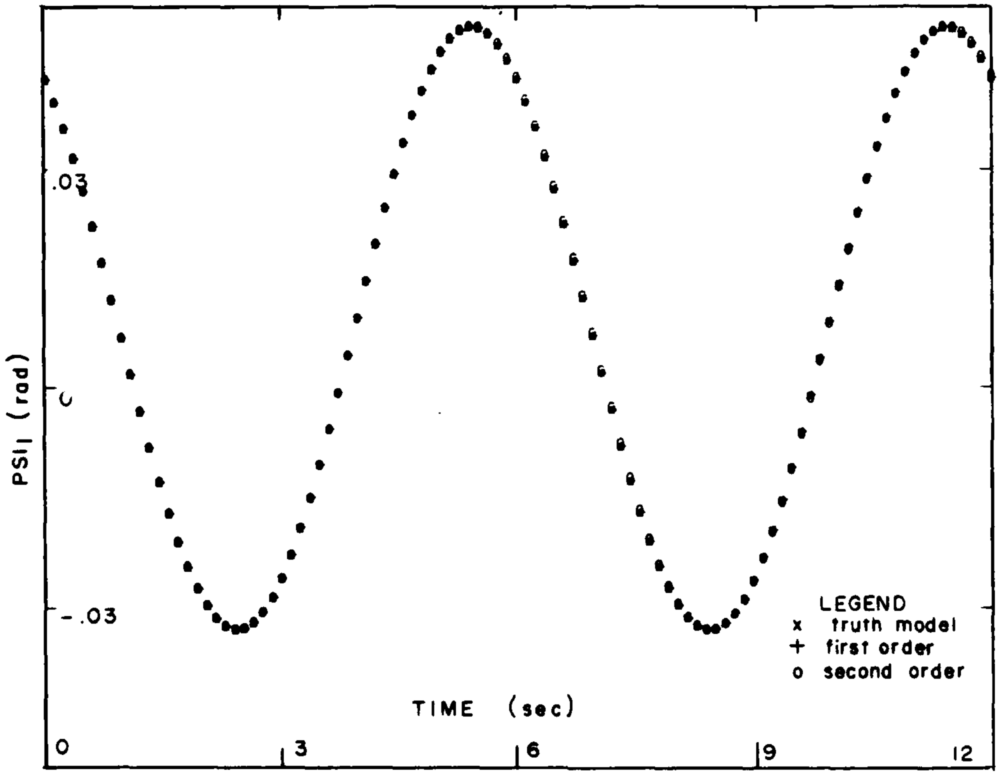


Figure 10. ψ_1 Single Scan Dynamics for Case 2 IC's

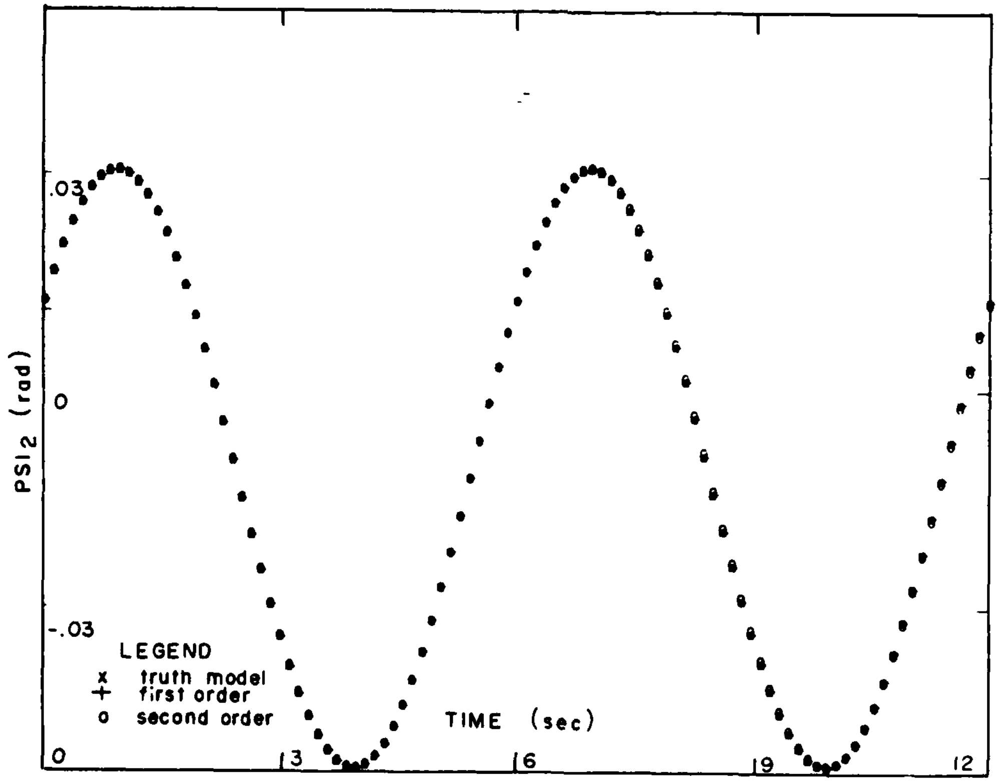


Figure 11. ψ_2 Single Scan Dynamics for Case 2 IC's

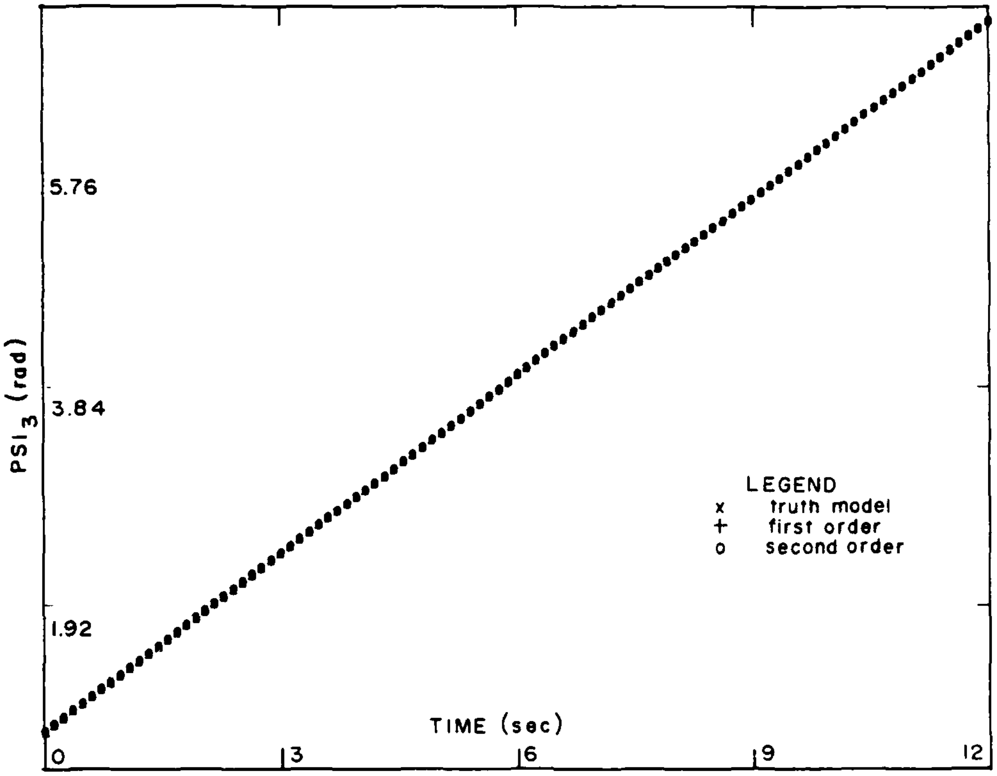


Figure 12. ψ_3 Single Scan Dynamics for Case 2 IC's

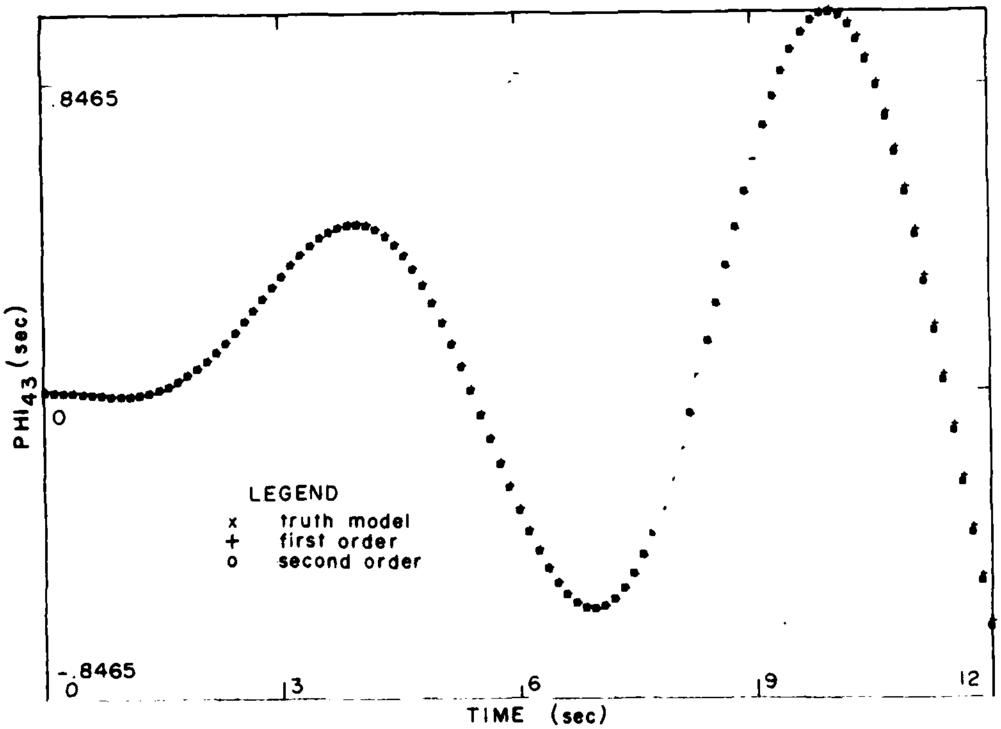


Figure 13. Single Scan Φ_{43} for Case 2 IC's

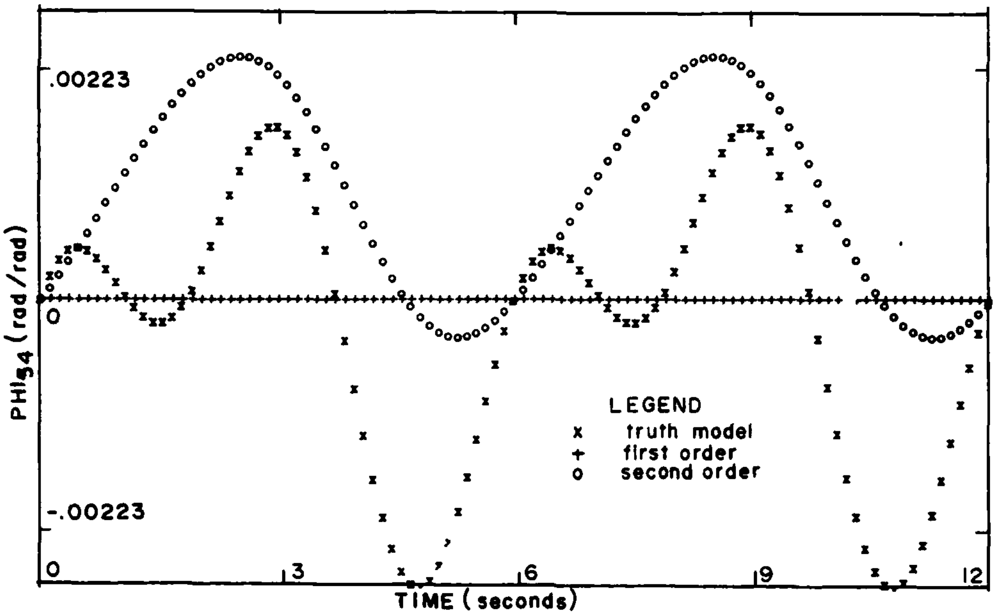


Figure 14. Single Scan Φ_{54} for Case 2 IC's

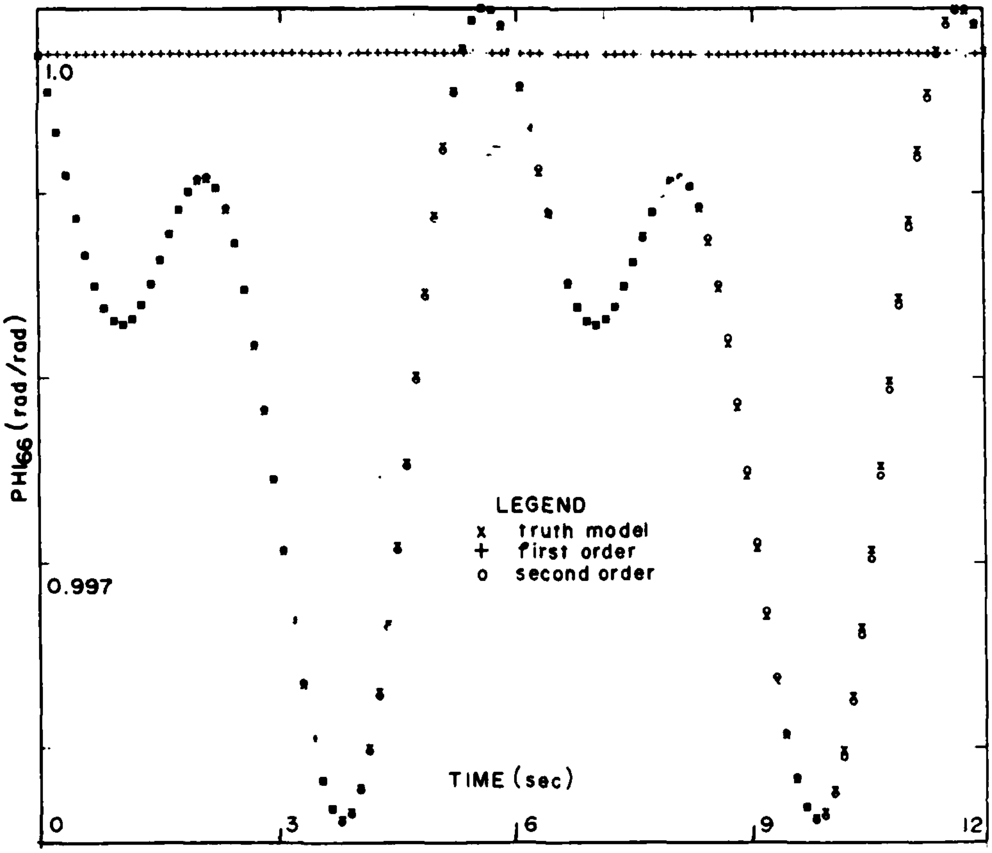


Figure 15. Single Scan Φ_{66} for Case 2 IC's

Test Case Results

Of the eight test cases which were analyzed, the first four cases contained no noise in the observation data. For the remaining test cases, pseudo-random normal noise was added to the truth model slit crossing times. The induced noise has a mean of 0 and one standard deviation equal to $3.16228 * 10^{-5}$ seconds (1 sigma).

Test Case 1 was run to analyze a single scan estimation with case 1 true initial conditions. The initial state estimate was set to the following values:

$\omega_{10}(\text{est}) = 0.011$ rad/sec

$\omega_{20}(\text{est}) = 0.055$ rad/sec

$\omega_{30}(\text{est}) = 0.5723598776$ rad/sec

$\psi_{10}(\text{est}) = 0.055$ rad

$\psi_{20}(\text{est}) = 0.055$ rad

$\psi_{30}(\text{est}) = 0.808$ rad

The estimation package converged after four iteration cycles. The converged estimate of the initial state reduced the total pointing error from 4.48 degrees to 10.335 arc seconds. Total pointing error is defined by the equation:

$$TPE = [(\delta\psi_{10})^2 + (\delta\psi_{20})^2 + (\delta\psi_{30})^2]^{1/2} \qquad (79)$$

A qualitative representation of the converged estimate is shown in Figure 16. This graph depicts a one scan propagation of the true and estimated dynamics of the star scanner boresight axis against the background star field. These results were felt to be quite adequate for a fine attitude estimator.

Test Case 2 is a repeat of test case 1, but with case 2 true initial conditions. This case converged to a total pointing error of 6.736 arc seconds. Results from this case are plotted in figure 17.

Test Case 3 was the final noise-free single scan estimation run performed. For this case, conditions were repeated from a starting value of 0 to -50 minutes, thus reworking Test Case 2 against a different star field. The converged total pointing error achieved was 40.09 arc seconds. Results from this case are plotted in Figure 18.

Test Case 4 checks the multiple scan capability of the estimation package. For this case, initial conditions identical to Test Case 1 were used. After achieving a converged state estimate at epoch, the approximate dynamics were then propagated for approximately 30 scans (5 minutes) before another estimation sequence was performed. This process was repeated for six estimation sequences. Results, showing total pointing error, are plotted in Figure 19. In this figure, it can be seen that the converged estimate for the fifth estimation sequence is substantially less accurate than the results achieved with the other estimation sequences. This reduced accuracy is due to the fact that only 5 stars were observed during the scan used to generate this estimation sequence, whereas the other sequences processed at least 10 stars.

The remaining test cases depicted realistic simulations with noise present in the observation data. Test Case 5 repeats Test Case 1, but with one sigma random normal noise added to the true observation times. Total pointing error for this case converged to 56.14 arc seconds (see Figure 20). Test Case 6 is a repetition of Test Case 1 with noise added. The total pointing error for this case converged to 57.15 arc seconds (see Figure 21). In this figure, the propagation of the initial estimate, as well as the converged estimate, are displayed.

Test Case 7 investigates the multiple scan capability of the estimator with short propagation periods (noise present). For this case, 10 estimation cycles were performed, with each estimation cycle separated by 60 seconds of state propagation. Initial conditions were identical to Test Case 1. Results show this type of estimation scheme to be good at maintaining high accuracy in total pointing angle (see Figure 22).

A final Test Case was run to investigate the long term capability of the estimator. Using identical conditions to test Case 4, results show that the presence of noise does not seriously degrade the estimator's performance. (see Figure 23).

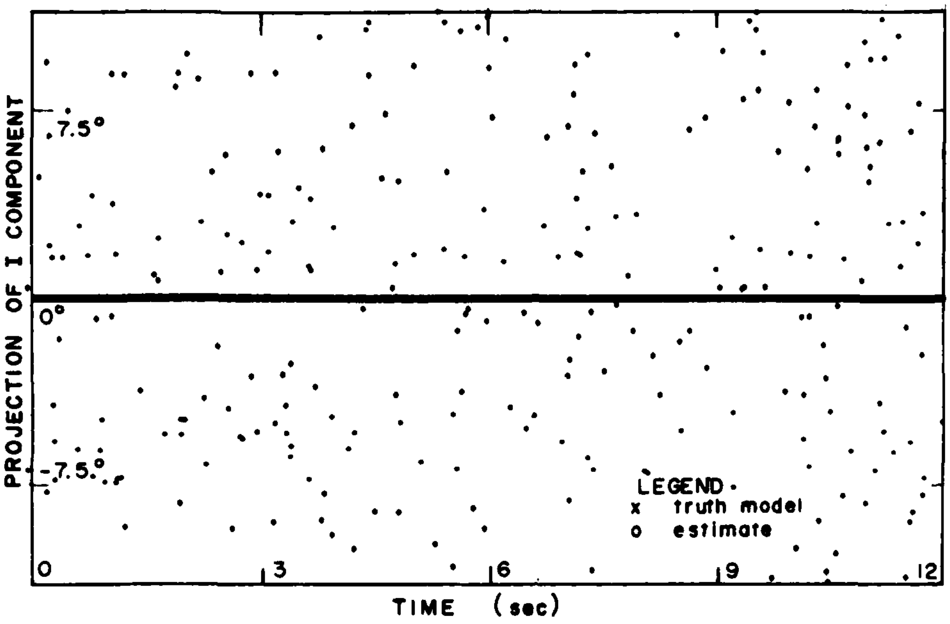


Figure 16. Test Case #1 State Propagation

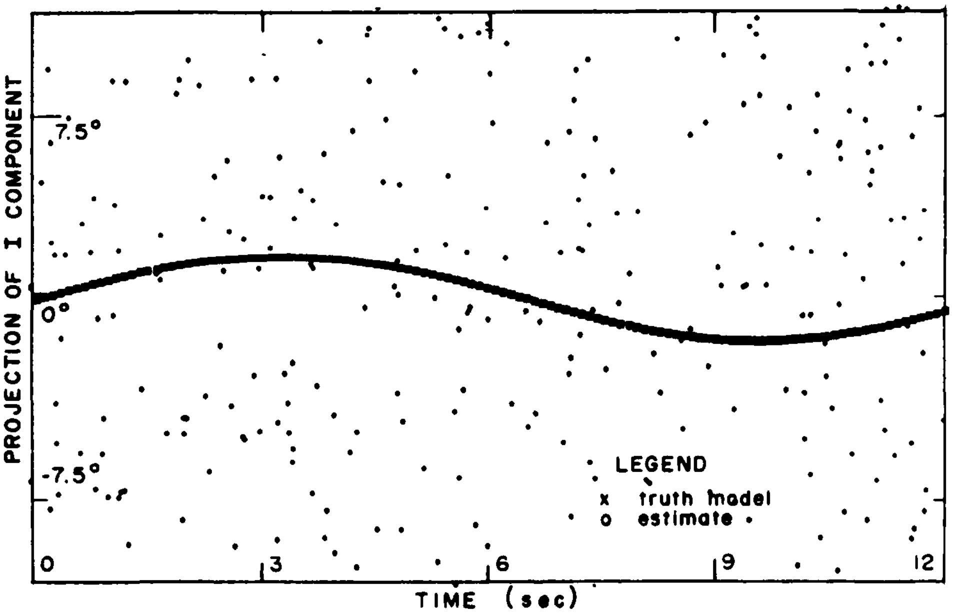


Figure 17. Test Case #2 State Propagation

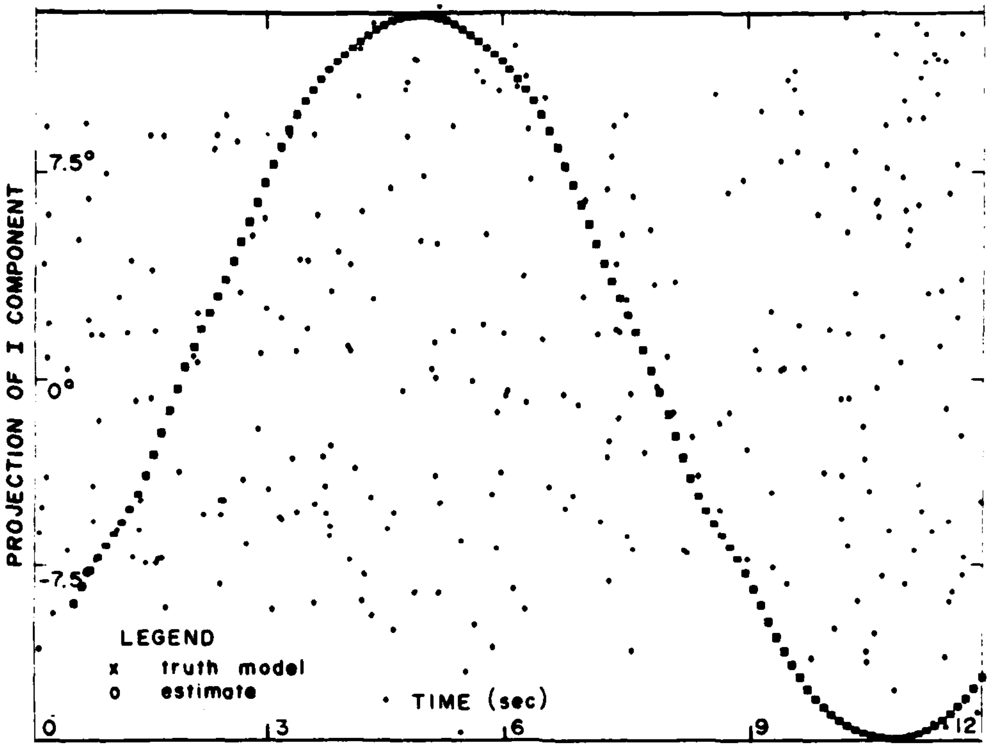


Figure 18. Test Case #3 State Propagation

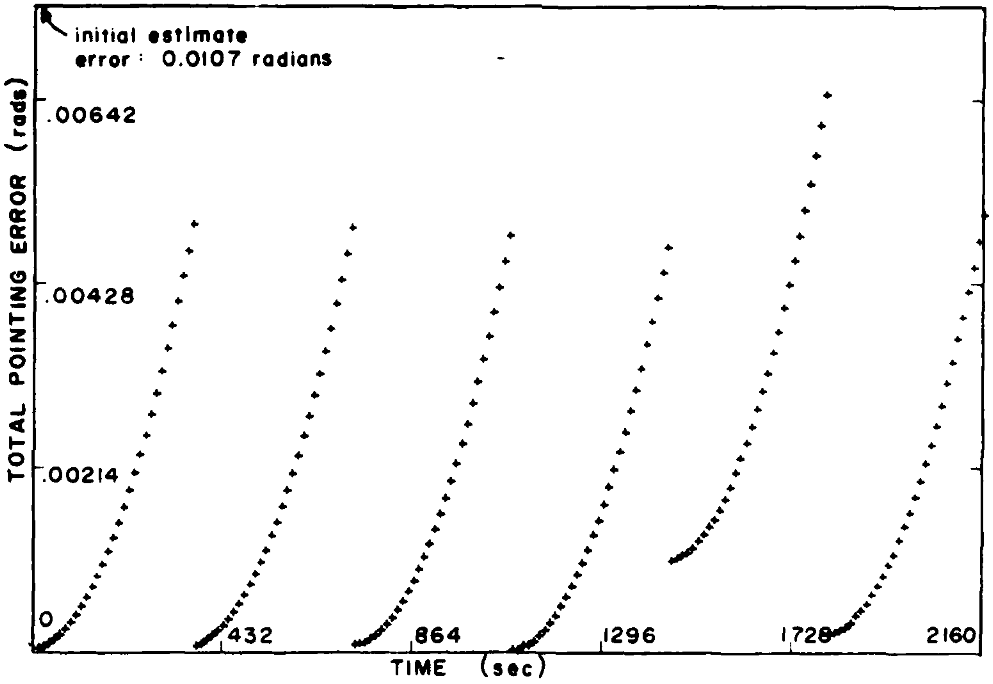


Figure 19. Test Case #4 State Total Pointing Error

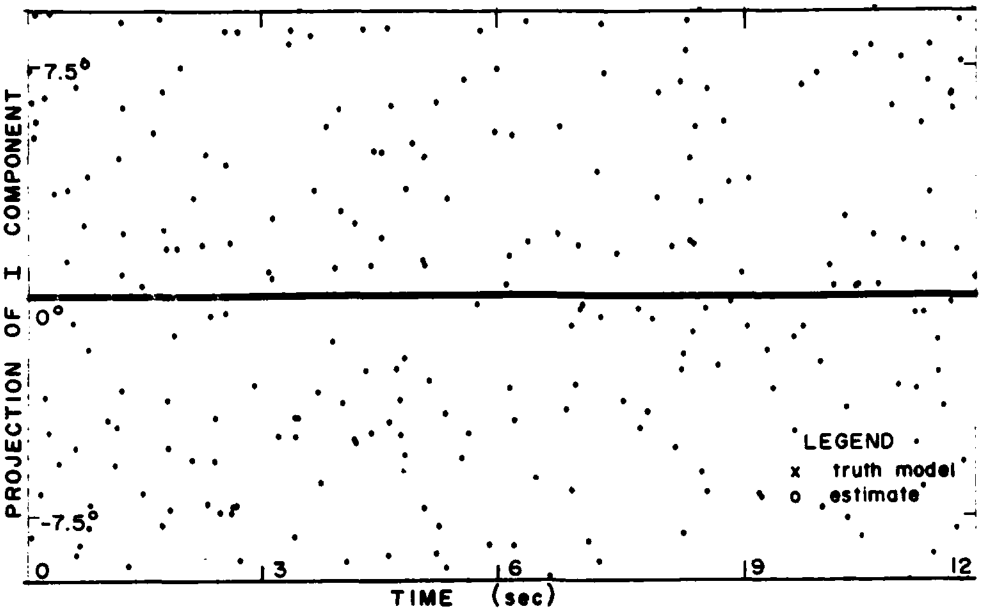


Figure 20. Test Case #5 State Propagation

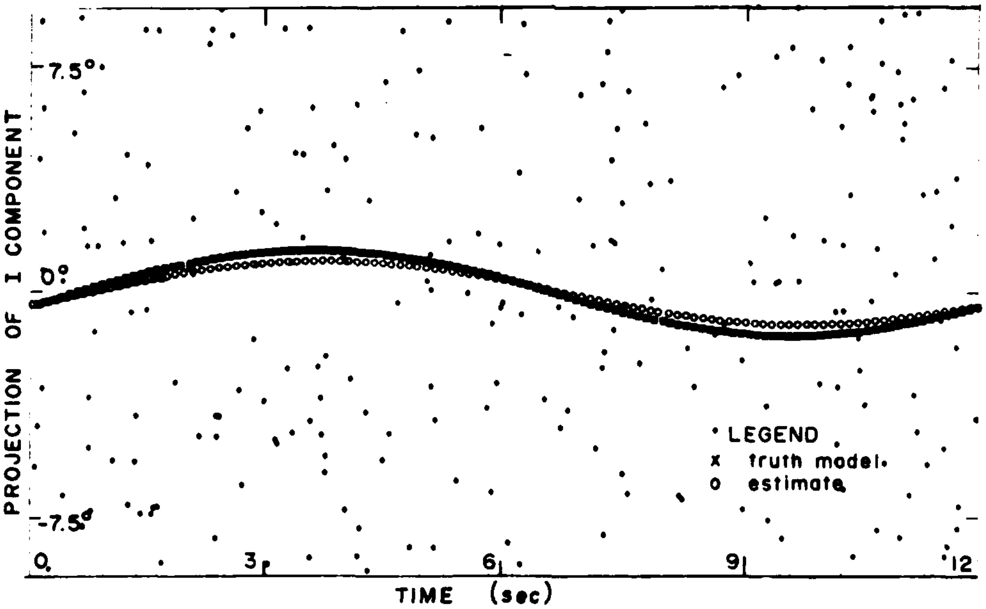


Figure 21. Test Case #6 State Propagation

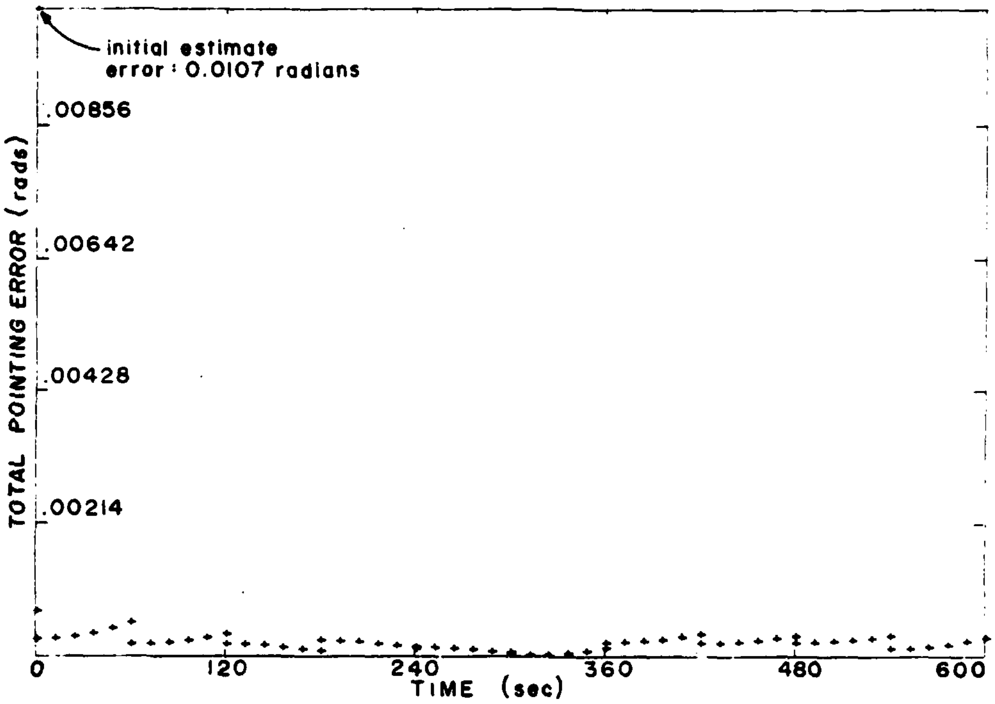


Figure 22. Test Case #7 State Total Pointing Error

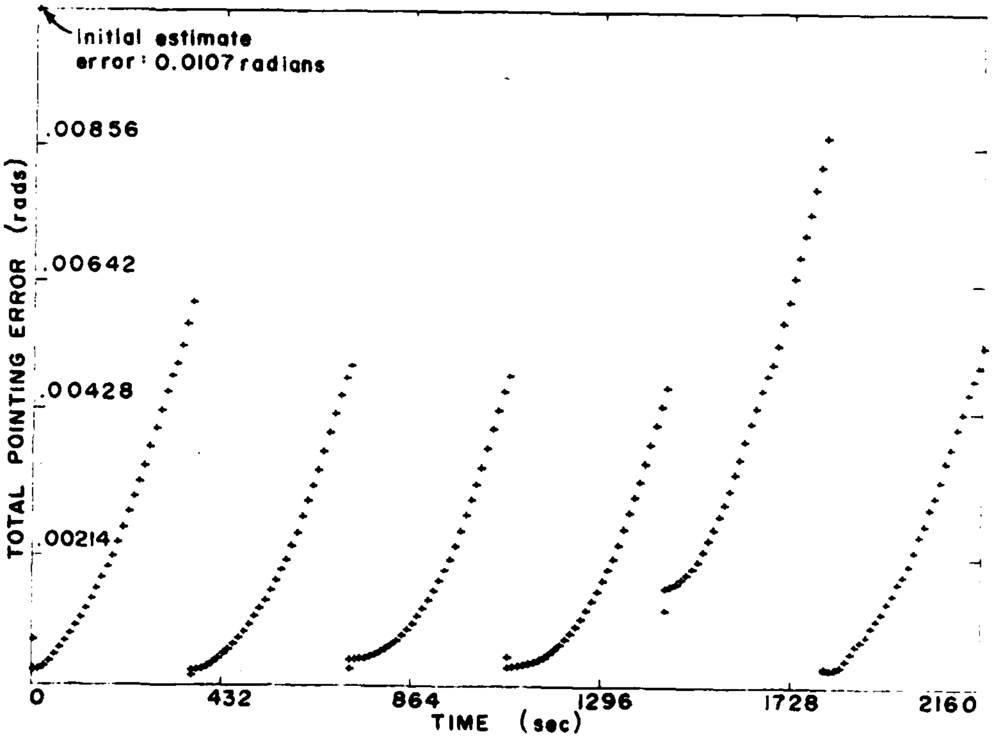


Figure 23. Test Case #8 State Total Pointing Error

Chapter Five Conclusions

The goal of this research was the development of a fine attitude estimation routine for a torque-free axisymmetric spinning satellite. The results obtained in the previous section demonstrate that the model developed in this research, when run with short propagation cycles, can adequately predict the satellite model torque-free attitude state.

Figure 22 shows that with estimation updates each 60 seconds, the total pointing error was reduced from an initial error of 0.0107 radians and maintained below a maximum total pointing error of 0.0004 radians. This result equates to a ground distance error of approximately nine statute miles from geosynchronous altitude, and it should prove suitable for most pointing requirements. Figure 23 shows what happens when the estimation update time is extended to five minutes. At the end of the the last propagation cycle in this figure, the total pointing error has grown to approximately 0.00544 radians, or about 120 statute miles ground error from geosynchronous altitude. This error is about half as large as the initial estimate error, and it might present an outer bound for estimate update cycle times. The model was not tested for large initial estimate errors, and further investigation in this area seems warranted. Also, the package developed might not be adequate for an environment where continous torques are present, but small, impulsive torques, such as those generated by fine attitude correction jets, could be handled by the torque-free model.

The attitude prediction capability of this estimation package was shown to be degraded in the presence of a low density star field. This problem arises from the method chosen to govern the estimation sequence. A possible alternative to this scheme would be to base the estimation sequence on a predetermined number of observations, rather than the current method, which searches for a specified time period and uses all data recorded during that period.

Chapter Six <u>Recommendations for Follow-On Analysis</u>

The author feels there are several areas where the work presented in this paper could be used as a basis for follow-on research. The first area of investigation which presents itself would be the development of a star identification system. This addition would establish a complete attitude estimation package.

Another area for follow-on research might be an investigation of alternative estimation routines. As stated earlier, the capability of this estimator is seriously degraded in the presence of a low density star field. This drawback could be overcome by the choice of a different basis for the least-squares estimation cycle. Another alternative might be the use of a Bayes estimator instead of the least-squares algorithm. Still another alternative might be the use of a Kalman filter to replace or augment the batch estimator.

A final recommended follow-on research would be an investigation of the use of this estimation package as a controller for some attitude correction package. The result of such an effort could be the presentation of an attitude control system.

BIBLIOGRAPHY

1. Proceedings of the Symposium on Spacecraft Attitude Determination, September 30,October 1-2,1969, Vol. I,Air Force Report No. SAMSO-TR-69-417.

2. Grosch, Charles B."Orientation of a Rigid Torque-Free Body by Use of Star Transits",Journal of Spacecraft,Vol. 4, Num. 5,562-566,May 1967

3. Gutshall, R.L., and Deters, R.A., "A Survey of Attitude Sensors", The Journal of the Astronautical Sciences, Vol. XXVII, No. 3, 217-238, July-September, 1979

4. U.S. Naval Observatory. The Astronautical Almanac for the Year 1984. Washington : U.S. Government Printing Office, 1983.

5. Norton, Arthur P. Norton's Star Atlas and Reference Handbook (Epoch 1950.0) (Sixteenth Edition). Cambridge : Sky Publishing Corporation, 1973

6. Kane, Thomas R., Likins, Peter W., and Levinson, David A. Spacecraft Dynamics, McGraw Hill, Inc., New York, 1983

7. Wertz, James. Spacecraft Attitude Determination and Control, D. Reidel Publishing Company, Boston:U.S.A.,1983

8. Meirovitch, Leonard. Methods of Analytical Dynamics, McGraw-Hill, Inc., New York 1970

9. Wiesel, Dr. William E.,Professor, Air Force Institute of Technology,Wright Patterson AFB, OH. Personal Interview 27 AUG 1986

10. Bate, Roger R.,Mueller, Donald D., and White, Jerry E. Fundamentals of Astrodynamics, Dover Publications, Inc., New York, 1971

11. Wiesel, Dr. William E., Modern Methods of Orbit Determination, Air Force Institute of Technology, WPAFB, OH, 1985

12. Ward, Capt John E.Jr., Autonomous State Determination for an Earth Orbiting Satellite Using Horizon and Star Sensors. MS Thesis, School of Engineering, Air Force Institute of Technology (AU),WPAFB,OH, December, 1984

13. IMSL Library Reference Manual, IMSL, Inc., Houston, TX, 1984

APPENDIX A

Tables

Table 1 BRIGHT STAR DATA BASE

7epsAur1605	5	00	51.2	43	48	05	2.99
8zetAur1612	5	01	23.5	41	03	15	3.75
10betCam1603	5	02	02.1	60	25	16	4.03
102iotTau1602	5	02	10.1	21	34	07	4.64
11 Ori1638	5	03	40.9	15	23	00	4.68
etaPic1663	5	04	33.8	-49	35	54	5.03
2epsLep1654	5	04	48.2	-22	23	28	3.19
zetDor1674	5	05	14.6	-57	29	37	4.72
10etaAur1641	5	05	25.5	41	12	53	3.17
67betEri1666	5	07	05.2	-05	06	21	2.79
69lamEri1679	5	08	24.2	-08	46	24	4.27
16 Ori1672	5	08	28.4	09	48	38	5.43
3iotLep1696	5	11	34.4	-11	53	13	4.45
5mu Lep1702	5	12	14.0	-16	13	24	3.31
11mu Aur1689	5	12	21.9	38	28	02	4.86
17rhoOri1698	5	12	28.8	2	50	37	4.46
4kapLep1705	5	12	30.9	-12	57	33	4.36
theDor1744	5	13	46.0	-67	12	10	4.83
19betOri1713	5	13	47.5	-08	13	08	0.12
13alpAur1708	5	15	32.5	45	59	00	0.08
20tauOri1735	5	16	51.2	-06	51	37	3.60
omiCol1743	5	16	55.5	-34	54	36	4.83
15lamAur1729	5	18	02.9	40	05	11	4.71
6lamLep1756	5	18	51.6	-13	11	31	4.29
zetPic1767	5	18	59.2	-50	37	20	5.45
22 Ori1765	5	20	58.2	-00	23	49	4.73
29 Ori1784	5	23	12.0	-07	49	18	4.14
28etaOri1788	5	23	41.8	-02	24	38	3.36
24gamOri1790	5	24	17.9	06	20	11	1.64
112betTau1791	5	25	18.7	28	35	43	1.65
115 Tau1808	5	26	15.8	17	56	59	5.42
9betLep1829	5	27	34.8	-20	46	16	2.84
1856	5	29	43.9	-47	05	18	5.46
32 Ori1839	5	29	57.3	05	56	14	4.20
epsCol1862	5	30	39.7	-35	28	53	3.87
34delOri1852	5	31	12.8	-00	18	35	2.23
119 Tau1845	5	31	18.2	18	35	01	4.38
25chiAur1843	5	31	43.1	32	10	54	4.76
11alpLep1865	5	32	02.7	-17	49	58	2.58
gamMen1953	5	32	29.6	-76	21	10	5.19
betDor1922	5	33	29.4	-62	30	00	3.40
37phiOri1876	5	33	58.1	09	28	48	4.41
39lamOri1879	5	34	17.0	09	55	29	3.39
44iotOri1899	5	34	40.4	-05	55	09	2.76
46epsOri1903	5	35	25.6	-01	12	40	1.70
40phiOri1907	5	36	03.2	09	16	59	4.09
123zetTau1910	5	36	43.1	21	08	03	3.00
48sigOri1931	5	37	58.0	-02	36	29	3.81
alpCol1956	5	39	05.2	-34	04	55	2.64
50zetOri1948	5	39	58.5	-01	57	00	1.77

13gamLep1983	5	43	49.0	-22	27	10	3.60
27omiAur1971	5	44	41.9	49	49	15	5.47
delDor2015	5	44	44.7	-65	44	29	4.35
14zetLep1998	5	46	15.1	-14	49	37	3.55
130 Tau1990	5	46	31.9	17	43	27	5.49
betPic2020	5	46	55.0	-51	04	18	3.85
53kapOri2004	5	47	01.2	-09	40	28	2.06
gamPic2042	5	49	32.7	-56	10	13	4.51
betCol2040	5	50	24.8	-35	46	25	3.12
32nu Aur2012	5	50	24.9	39	08	42	3.97
2049	5	50	32.2	-52	06	44	5.17
15delLep2035	5	50	39.3	-20	52	47	3.81
136 Tau2034	5	52	21.2	27	36	34	4.58
54chiOri2047	5	53	27.8	20	16	27	4.41
30xi Aur2029	5	53	32.8	55	42	17	4.99
58alpOri2061	5	54	19.9	07	24	18	0.50
16etaLep2085	5	55	41.9	-14	10	12	3.71
gamCol2106	5	56	59.2	-35	17	04	4.36
60 Ori2103	5	58	01.7	00	33	08	5.22
33delAur2077	5	58	15.0	54	17	05	3.72
34betAur2088	5	58	23.5	44	56	49	1.90
37theAur2095	5	58	39.8	37	12	45	2.62
etaCol2120	5	58	40.3	-42	48	56	3.96
35pi Aur2091	5	58	47.1	45	56	12	4.26
61mu Ori2124	6	01	31.8	09	38	54	4.12
62chiOri2135	6	02	59.9	20	08	23	4.63
1 Gem2134	6	03	10.7	23	15	55	4.16
17 Lep2148	6	04	17.6	-16	28	57	4.93
67nu Ori2159	6	06	41.2	14	46	16	4.42
2180	6	08	18.7	-22	25	26	5.50
nu Dor2221	06	08	50.3	-68	50	25	5.06
delPic2212	6	09	59.8	-54	57	53	4.81
alpMen2261	6	10	42.3	-74	44	54	5.09
70xi Ori2199	6	11	03.5	14	12	47	4.48
36 Cam2165	6	11	17.5	65	43	23	5.32
7etaGem2216	6	13	56.5	22	30	44	3.28
5gamMon2227	6	14	05.9	-06	16	09	3.98
44kapAur2219	6	14	23.4	29	30	17	4.35
74 Ori2241	6	15	34.4	12	16	39	5.04
kapCol2256	6	16	00.0	-35	08	06	4.37
2209	6	17	08.4	69	19	37	4.80
2 Lyn2238	6	18	15.4	59	01	05	4.48
7 Mon2273	6	18	58.0	-07	48	57	5.27
1zetCMa2282	6	19	43.1	-30	03	21	3.02
delCol2296	6	21	32.8	-33	25	41	3.85
2betCMa2294	6	22	01.0	-17	56	51	1.98
13mu Gem2286	6	22	01.4	22	31	21	2.88
8 Mon2298	6	22	56.8	04	36	06	4.33
2305	6	23	26.9	-11	31	16	5.22
alpCar2326	6	23	36.5	-52	41	13	-0.72
46psiAur2289	6	23	42.3	49	17	50	4.91
10 Mon2344	6	27	11.6	-04	45	06	5.05

lamCMa2361	6	27	35.6	-32	34	11	4.48
18nu Gem2343	6	28	02.6	20	13	22	4.15
4xi Cma2387	6	31	12.6	-23	24	24	4.34
13 Mon2385	6	32	03.9	07	20	43	4.50
2395	6	32	50.7	-01	12	28	5.10
5xi2Cma2414	6	34	24.4	-22	57	07	4.54
2435	6	34	38.0	-52	57	45	4.39
7nu2CMa2429	6	36	00.4	-19	14	32	3.95
24gamGem2421	6	36	49.0	16	24	48	1.93
8nu3CMa2443	6	37	12.5	-18	13	24	4.43
nu Pup2451	6	37	17.2	-43	10	55	3.17
15 Mon2456	6	40	07.5	09	54	39	4.65
27epsGem2473	6	42	58.7	25	08	51	2.98
30 Gem2478	6	43	06.9	13	14	40	4.49
2401	6	43	36.6	79	35	03	5.45
31xi Gem2484	6	44	25.2	12	54	47	3.36
9alpCMa2491	6	44	27.9	-16	41	39	-1.46
56psiAur2483	6	45	37.4	43	35	38	5.25
57psiAur2487	6	46	28.7	48	48	26	5.22
2518	6	46	49.5	-37	54	44	5.26
18 Mon2506	6	47	03.1	02	25	48	4.47
alpPic2550	6	48	02.0	-61	55	29	3.27
13kapCMa2538	6	49	15.7	-32	29	24	3.96
2554	6	49	31.1	-53	36	13	4.40
tauPup2553	6	49	33.1	-50	35	45	2.93
iotVol2602	6	51	37.8	-70	56	40	5.40
34theGem2540	6	51	46.1	33	58	51	3.60
43 Cam2511	6	52	02.3	68	54	29	5.12
14theCma2574	6	53	28.2	-12	01	07	4.07
16omiCMa2580	6	53	29.3	-24	09	50	3.86
20iotCMA2596	6	55	26.7	-17	02	00	4.38
15 Lyn2560	6	55	56.2	58	26	40	4.35
2527	6	57	49.3	76	59	58	4.55
21epsCMa2618	6	58	01.0	-28	57	01	1.50
285	1	06	25.0	86	10	28	4.25
1alpUMi 424	2	15	54.6	89	11	39	2.02
2609	7	33	39.1	87	03	21	5.07
3751	9	34	57.6	81	23	47	4.29
4084	10	29	16.6	82	38	18	5.26
22epsUMi6322	16	47	31.1	82	03	52	4.23
8546	22	14	36.9	86	01	50	5.27
8748	22	54	34.7	84	15	48	4.71
gamCam1148	3	48	42.2	71	17	10	4.63
9alpCam1542	4	52	30.1	66	19	05	4.29
7 Cam1568	4	56	02.4	53	43	43	4.47
30 Oph6318	17	00	14.5	-04	12	01	4.82
59 Her6332	17	01	02.0	33	35	24	5.25
60 Her6355	17	04	39.5	12	45	41	4.91
22zetDra6396	17	08	44.4	65	44	01	3.17
35etaOph6378	17	09	29.3	-15	42	24	2.43
etaSco6380	17	11	02.5	-43	13	11	3.33
64alpHer6406	17	13	56.4	14	24	26	3.08

65delHer6410	17	14	23.7	24	51	25	3.14
67pi Her6418	17	14	30.4	36	49	34	3.16
6452	17	19	37.9	18	04	20	5.00
53nu Ser6446	17	19	57.3	-12	49	55	4.33
40xi Oph6445	17	20	04.4	-21	05	50	4.39
72 Her6458	17	20	04.7	32	29	13	5.39
iotAps6411	17	20	21.9	-70	06	31	5.41
42theOph6453	17	21	03.4	-24	59	06	3.27
betAra6461	17	24	00.6	-55	30	59	2.85
gamAra6462	17	24	05.2	-56	21	52	3.34
44 Oph6486	17	25	25.4	-24	09	43	4.17
49sigOph6498	17	25	44.7	04	09	11	4.34
6493	17	25	48.5	-05	04	26	4.54
45 Oph6492	17	26	21.8	-29	51	15	4.29
delAra6500	17	29	41.9	-60	40	20	3.62
34nu Sco6508	17	29	42.5	-37	17	04	2.69
23betDra6536	17	30	04.9	52	18	45	2.79
76lamHer6526	17	30	06.7	26	07	18	4.41
alpAra6510	17	30	38.6	-49	51	55	2.95
24nu1Dra6554	17	31	52.2	55	11	40	4.88
25nu2Dra6555	17	31	57.6	55	10	59	4.87
27 Dra6566	17	32	01.5	68	08	42	5.05
35lamSco6527	17	32	33.3	-37	05	37	1.63
55alpOph6556	17	34	12.9	12	34	14	2.08
6546	17	35	28.7	-38	37	32	4.29
theSco6553	17	36	12.3	-42	59	21	1.87
55xi Ser6561	17	36	41.9	-15	23	23	3.54
28omeDra6596	17	37	02.4	68	45	55	4.80
23delUMi6789	17	37	10.7	86	35	45	4.36
85iotHer6588	17	39	01.6	46	00	51	3.80
56omiSco6581	17	40	32.6	-12	52	05	4.26
kapSco6580	17	41	24.9	-39	01	23	2.41
31psiDra6636	17	42	12.7	72	09	24	4.58
58 Oph6595	17	42	30.0	-21	40	36	4.87
60betOph6603	17	42	42.4	04	34	23	2.77
84 Her6608	17	42	43.4	24	20	02	5.71
mu Ara6585	17	42	54.8	-51	49	37	5.15
etaPav6582	17	44	12.7	-64	43	04	3.62
86mu Her6623	17	45	51.1	27	43	45	3.42
iotSco6615	17	46	30.0	-40	07	20	3.03
3chiSgr6616	17	46	35.1	-27	49	33	4.20
62gamOph6629	17	47	06.9	02	42	44	3.75
6630	17	48	48.2	-37	02	22	3.21
35 Dra6701	17	50	08.5	76	57	56	5.04
32zetDra6688	17	53	15.6	56	52	29	3.75
89 Her6685	17	54	47.6	26	03	07	5.46
91theHer6695	17	55	43.3	37	15	07	3.86
33gamDra6705	17	56	14.7	51	29	25	2.23
92chiHer6703	17	57	09.7	29	14	56	3.70
94nu Her6707	17	57	54.6	30	11	24	4.41
64nu Oph6698	17	58	10.4	-09	46	22	3.34
93 Her6713	17	59	22.0	16	45	04	4.67

67 Oph6714	17	59	52.1	02	55	53	3.97
68 Oph6723	18	00	58.0	01	18	17	4.45
W Sgr6742	18	04	01.8	-29	34	54	4.30
70 Oph6752	18	04	40.3	02	30	08	4.03
10gamSgr6746	18	04	48.7	-30	25	31	2.99
theAre6743	18	05	25.4	-50	05	38	3.66
72 Oph6771	18	06	36.9	09	33	39	3.73
103omiHer6779	18	06	56.3	28	45	35	3.83
6791	18	07	00.7	43	27	34	5.00
pi Pav6745	18	07	05.3	-63	40	14	4.35
102 Her6787	18	08	05.7	20	48	41	4.36
epsTel6783	18	10	04.7	-45	57	29	4.53
13mu Sgr6812	18	12	50.2	-21	03	50	3.86
36 Dra6850	18	13	48.4	64	23	31	5.03
6819	18	15	49.2	-56	01	46	5.33
etaSgr6832	18	16	34.7	-36	46	03	3.11
1kapLyr6872	18	19	19.1	36	03	25	4.33
19delSgr6859	18	20	00.1	-29	50	09	2.70
74 Oph6866	18	20	05.6	03	22	10	4.86
58etaSer6869	18	20	30.4	-02	54	13	3.26
43phiDra6920	18	20	58.8	71	19	48	4.22
44chiDra6927	18	21	20.2	72	43	35	3.57
xi Pav6855	18	21	48.0	-61	30	09	4.36
109 Her6895	18	23	02.2	21	45	43	3.84
20epsSgr6879	18	23	08.6	-34	23	35	1.85
39 Dra6923	18	23	41.0	58	47	29	4.98
alpTel6897	18	25	49.5	-45	58	41	3.51
22lamSgr6913	18	27	00.9	-25	25	52	2.81
zetTel6905	18	27	38.3	-49	04	50	4.13
gamSct6930	18	28	18.8	-14	34	36	4.70
60 Ser6935	18	28	52.6	-01	59	46	5.39
theCra6951	18	32	23.8	-42	19	29	4.64
alpSct6973	18	34	21.8	-08	15	21	3.85
6985	18	35	43.4	09	06	34	5.39
3alpLyr7001	18	36	24.8	38	46	07	0.03
zetPav6982	18	41	14.0	-71	26	36	4.01
delSct7020	18	41	25.5	-09	04	06	4.72
epsSct7032	18	42	40.6	-08	17	29	4.90
6zetLyr7056	18	44	14.3	37	35	18	4.36
27phiSgr7039	18	44	41.3	-27	00	28	3.17
110 Her7061	18	44	59.7	20	31	51	4.19
7064	18	45	27.0	26	38	42	4.83
chiOct6721	18	45	55.3	-87	37	27	5.28
111 Her7069	18	46	20.2	18	09	49	4.36
betSct7063	18	46	21.1	-04	45	55	4.22
R Sct7066	18	46	39.3	-05	43	21	5.20
50 Dra7124	18	46	52.6	75	24	59	5.35
etaCra7062	18	47	43.4	-43	41	53	5.49
10betLyr7106	18	49	30.4	33	20	39	3.45
lamPav7074	18	50	47.1	-62	12	25	4.22
47omiDra7125	18	50	58.4	59	22	09	4.66
12delLyr7139	18	53	57.7	36	52	43	4.30

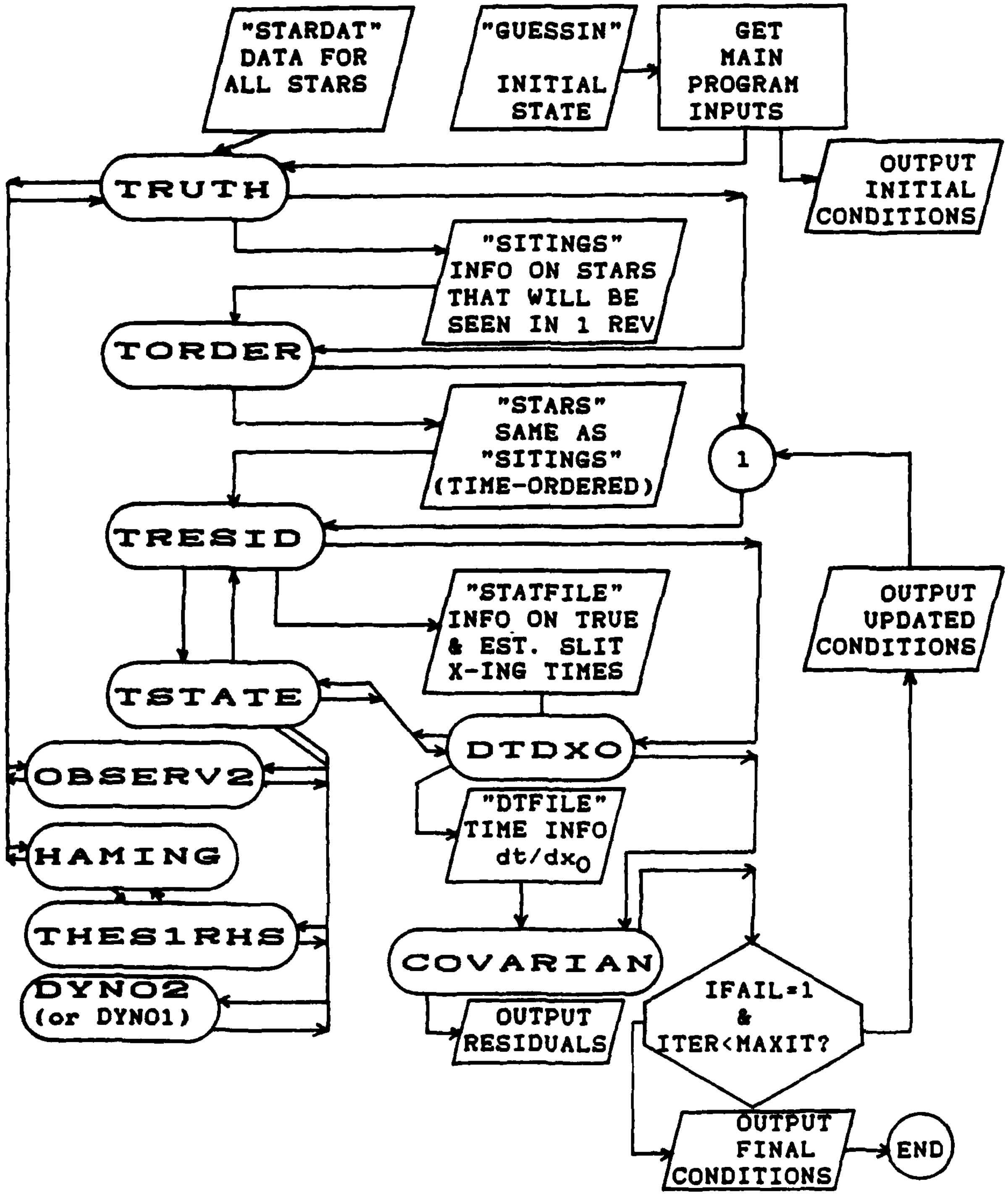
34sigSgr7121	18	54	18.3	-26	19	01	2.02
52nu Dra7180	18	54	35.5	71	18	36	4.82
13R Lyr7157	18	54	51.8	43	55	31	4.04
kapPav7107	18	55	21.5	-67	15	16	3.90
63theSer7141	18	55	26.9	04	10	58	4.06
37xi2Sgr7150	18	56	48.3	-21	07	41	3.51
lamTel7134	18	57	13.5	-52	57	36	5.03
14gamLyr7178	18	58	21.8	32	40	04	3.24
13epsAql7176	18	58	55.1	15	02	47	4.02
1kapCep7750	20	09	25.9	77	39	55	4.39
73 Dra7879	20	31	43.4	74	54	06	5.20
60tauDra7352	19	15	51.3	73	19	37	4.45
57delDra7310	19	12	33.2	67	38	04	3.07
61sigDra7462	19	32	23.6	69	38	05	4.68
63epsDra7582	19	48	13.8	70	13	43	3.83
21etaUMi6116	16	17	56.7	75	47	29	4.95
15 Dra6161	16	28	00.5	68	48	06	5.00
16zetUMi5903	15	44	35.3	77	50	34	4.32
4 UMi5321	14	08	53.5	77	37	13	4.82
mu Hyi0776	2	31	58.7	-79	10	37	5.28
gamHyi1208	3	47	28.2	-74	17	12	3.24
etaMen1629	4	55	37.5	-74	57	40	5.47
gamVol2735	7	08	50.4	-70	28	20	3.62
delVol2803	7	16	50.5	-67	55	45	3.98
zetVol3024	7	42	01.1	-72	34	09	3.95
zetOct3678	8	59	09.1	-85	36	10	5.42
delCha4234	10	45	38.7	-80	27	30	4.45
iotOct4870	12	53	14.2	-85	02	22	5.46
alpAps5470	14	45	53.2	-78	58	49	3.83
delAps6020	16	17	59.7	-78	39	32	4.68
gamAps6102	16	31	02.6	-78	51	52	3.89
betAps6163	16	40	50.4	-77	29	14	4.24
alpTrA6217	16	47	01.0	-69	00	03	1.92
etaAra6229	16	48	26.5	-59	00	54	3.76
epsPav7590	19	58	49.2	-72	57	11	3.96
sigOct7228	20	54	13.8	-89	01	05	5.47
alpOct8021	21	02	52.7	-77	05	03	5.15
epsOct8481	22	18	21.1	-80	31	03	5.10
betOct8630	22	44	32.2	-81	27	48	4.15
tauOct8862	23	26	07.4	-87	34	04	5.49
alpCha3318	08	18	56.7	-76	52	16	4.07
theCha3340	08	21	07.7	-77	26	06	4.35
nu Cha3502	08	41	52.9	-78	54	27	5.47

APPENDIX B

Main Program

Listings

MAIN PROGRAM ESTMATOR.F


"STARDAT" DATA FOR ALL STARS
"GUESSIN" INITIAL STATE
GET MAIN PROGRAM INPUTS
OUTPUT INITIAL CONDITIONS
TRUTH
"SITINGS" INFO ON STARS THAT WILL BE SEEN IN 1 REV
TORDER
"STARS" SAME AS "SITINGS" (TIME-ORDERED)
1
TRESID
"STATFILE" INFO ON TRUE & EST. SLIT X-ING TIMES
OUTPUT UPDATED CONDITIONS
TSTATE
DTDXO
OBSERV2
"DTFILE" TIME INFO dt/dx0
HAMING
THES1RHS
COVARIAN
IFAIL=1 & ITER<MAXIT?
DYNO2 (or DYNO1)
OUTPUT RESIDUALS
OUTPUT FINAL CONDITIONS
END

PROGRAM LISTING   estmator.f

```
      program estmater
c     This is the main program shell
      common/sidereal/tgo
      common/dyncom/xges0(6),a(6)
      common/ham/t,xtrue(12,4),f(12,4),errest(12),n,h
      common/cov/q
      double precision x0(6),x(6),t,h,tend,t0,xges(6),xdel(6),tgo
      double precision a,f,errest,xtrue,xges0,tsince,tnew,tnow,q
      double precision tseed
      open(1,file='guessin')
c     read in number of odes,no. of epochs, no. of scans between update
      read(1,104) n,nepoch,nscan
c     read in start time,timestep, and rel. sidereal time
      read(1,105) t,h,tgo
c     specify an interval to be scanned and initialize time parms
      tend=12.00d+00
      t0=t
      tsince=t
      tnow=t-tgo
c     read initial conditions
      read(1,106) (x0(i),i=1,n)
c     write out initial conditions
      write(*,1)
      write(*,2) t,(i,x0(i),i=1,n)
c     read in initial guess
      read(1,100) maxit
  100 format(1x,i2)
      do 30 i=1,6
      read(1,101) xges(i)
   30 continue
  101 format(1x,e20.13)
c     read q (sigma squared)
      read(1,101) q
c     read noise flag (if flag=0 no noise will be added)
      read(1,100) inoise
c     read noise seed
      read(1,101) tseed
      close(1)
      write(*,107) q,tgo
      if(inoise.eq.0) write(*,108)
      if(inoise.ne.0) write(*,109) tseed
c     open file (errlist) for writing errors
      open(3,file='errlist')
      do 300 iloop=1,nepoch
      iter=1
      do 45 ii=1,6
      xges0(ii)=xges(ii)
   45 continue
c     write out estimate at epoch
      write(*,40)
   40 format(12x,'Initial estimate at epoch :')
      write(*,41) t,(iges,xges(iges),iges=1,6)
   41 format(8x,'t0 :',3x,e20.13,/,
     +    6(8x,'xges',i1,'0 :',3x,e20.13,/))
c     write out investigation time info
      write(*,111) nepoch,nscan
  111 format(x,///,8x,'This run covers ',i5,' epoch updates',/,
     + 8x,'with ',i5,' scans between each update')
      call cmperr(x0,xges,tgo,tperr,rtgo)
      write(3,103) rtgo,tperr
      write(*,4)
      call truth (t0,tend,h,x0,t,x)
      tnew=t
c     now order truth data by time
      write(*,5)
      call order(inoise,tseed)
c     get time residuals
   10 continue
      write(*,6)
      call resid(t0,xges)
c     get T
      write(*,7)
      call dtdxo(t0,xges)
c     get state corrections
      write(*,8)
      call covar(xdel,iter,ifail)
c     correct the guess
      do 9 i=1,6
      xges(i)=xges(i)-xdel(i)
    9 continue
c     write out updated guess
      write(*,42) tnow,(j,xges(j),xdel(j),j=1,6)
   42 format(2x,'EST Corrections and Updated ESTIMATE',/,
     + 4x,'tepoch :',3x,e20.13,/,
     + 6(4x,'xges',i1,'0 :',3x,e20.13,' from xdel of :',3x,
     + e20.13,/))
      call cmperr(x0,xges,tgo,tperr,rtgo)
      write(3,103) rtgo,tperr
c     if no convergence  go again
      iter=iter+1
      if(iter.le.maxit.and.ifail.eq.1) goto 10
      if(ifail.eq.1) write(*,102) maxit
  102 format(2x,'PROGRAM FAILED AFTER',i2,'ITERATIONS EXCEEDED')
c     At this point we have an estimate for epoch state
c     and the true state at tepoch and tend
c     now need to bring the estimate forward to tend
c     first   update times
      t=tnew-tend
      tgo=tgo-tend
      tsince=tsince+tend
c     set estimated state to least square converged value
      do 50 ip=1,6
      xges0(ip)=xges(ip)
   50 continue
      call approx2(tsince,t0)
c     write out final conditions
      write(*,3) tnew,tsince,(i,x(i),a(i),i=1,6)
      call cmperr(x,a,tgo,tperr,rtgo)
      write(3,103) rtgo,tperr
c     if there are more than one scans between epoch updates,
c     perform operations to bring true state and state estimate
c     forward to new epoch
      if(nscan.eq.1) goto 320
c     inner loop
      do 310 jloop=1,nscan-1
c     initialize haming
      nxt=0
      do 330 k=1,6
      xtrue(k,1)=x(k)
  330 continue
      call haming(nxt)
      if(nxt.ne.0) goto 340
      write(*,341)
  341 format(2x,'HAMING WOULD NOT INITIALIZE')
      stop
  340 continue
  360 continue
      call haming(nxt)
      if(t.le.tend) goto 360
c     update times
      t=t-tend
      tgo=tgo-tend
      tsince=tsince+tend
      do 350 ji=1,6
      x(ji)=xtrue(ji,nxt)
  350 continue
c     bring state estimate forward
      call approx2(tsince,t0)
c     write state and state estimate
      write(*,345) tgo,tsince,(xtrue(ii,nxt),a(ii),ii=1,6)
  345 format(2x,'State propogation',/,5x,'sidereal time :  ',e20.13,
     + 5x,'time since epoch : ',e20.13,/,20x,'STATES',/,
     + 6(5x,e20.13,5x,e20.13,/))
      call cmperr(x,a,tgo,tperr,rtgo)
      write(3,103) rtgo,tperr
  310 continue
  320 continue
c     now reset parameters for new epoch
      do 43 i=1,6
      xges(i)=a(i)
      x0(i)=xtrue(i,nxt)
   43 continue
      t0=t
      tsince=0.00d+00
      tnow=t-tgo
  300 continue
c     put tflag at end of errlist and close errlist
      rtgo=86400.0
      write(3,103) rtgo,tperr
  103 format(2x,f15.7,2x,f15.7)
      close(3)
c     now call ploterr which puts pointing errors into errplot
      call plterr
    1 format(2x,////,30x,'MAIN OUTPUT',/)
    2 format(8x,'Initial conditions :',/,
     +  8x,'t0 :',3x,e20.13,/,
     +  6(7x,'x',i1,'0 :',3x,e20.13,/))
    3 format(8x,'Final conditions :',/,
     + 19x,'true state',13x,'state estimate',/,
     +  8x,'t :',3x,e20.13,'test:',e20.13,/,
     +  6(7x,'x',i1,' :',3x,e20.13,5x,e20.13,/))
    4 format(1x,'ENTERING TRUTH')
    5 format(1x,'ENTERING ORDER')
    6 format(1x,'ENTERING RESID')
    7 format(1x,'ENTERING DTDXO')
    8 format(1x,'ENTERING COVAR')
  104 format(2x,i1,2x,i3,2x,i3)
  105 format(1x,e20.13,/,1x,e20.13,/,1x,e20.13)
  106 format(1x,e20.13,/,4(1x,e20.13,/),1x,e20.13)
  107 format(8x,'covariance : ',e20.13,/,8x,'initial tgo : ',e20.13,/)
  108 format(8x,'NO NOISE ADDED')
  109 format(8x,'NOISE rnd SEED : ',e20.13)
      stop
      end


      include'truth'
      include'torder'
      include'tstate'
      include'dtdx0'
      include'tresid'
      include'covarian'
      include'comperr'
      include'ploterr'
```

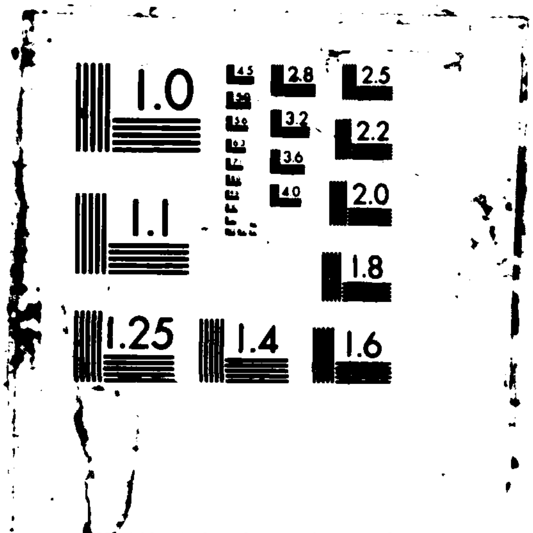
PROGRAM LISTING   comperr

```
      subroutine cmperr(xtrue,xest,tgo,tperr,time)
c     this routine takes the state vectors (true and estimated)
c     and computes the total pointing error
c     error information is temporarily held in file (errlist)
c
      double precision xtrue(6),xest(6)
      dimension x(6),xg(6),err(3)
      do 1 i=1,6
      x(i)=xtrue(i)
      xg(i)=xest(i)
    1 continue
      time=tgo
      do 2 i=1,3
      err(i)=x(i+3)-xg(i+3)
    2 continue
      tperr=sqrt(err(1)**2+err(2)**2+err(3)**2)
      return
      end
```

PROGRAM LISTING covarian

```
      subroutine cover(delx,jrej,ifail)
c     This routine reads the residual info and T from dtfile and
c     after computing the covariance, computes guess corrections
      common/cov/q
      double precision q,tres,Tval(6),ttrue,tst,qinv,rejec,tqmag
      double precision pinv(6,6),p(6,6),delx(6),tqinvr(6),tmat(6,6)
      dimension work(72)
c     constants
      qinv=1.00d+00/q
c     set rejection criterion
c     if this is less than the third iteration, rejec set to
c     3 seconds (1E05*sigma)
c     if third or higher iteration, rejec set to 3*sigma
c     if all residuals are rejected, rejec flag is widened
      rejec=3.00d+00*dsqrt(q)
      if(jrej.lt.3) rejec=3.00d+00
   30 continue
c     open file dtfile for reading
      open(1,file='dtfile')
c     initialize tqinvr vector and pinv matrix
      do 2 i=1,6
      tqinvr(i)=0.00d+00
      do 1 j=1,6
      pinv(i,j)=0.00d+00
    1 continue
    2 continue
      ifail=0
   10 continue
c     read in a data line
      read(1,100) islit,ttrue,tst
  100 format(7x,i1,2(2x,e20.13))
      if(islit.eq.0) goto 20
      read(1,101) (Tval(i),i=1,6)
  101 format(3(2x,e20.13),/,3(2x,e20.13))
c     compute residuals and print them
      tres=ttrue-tst
      irej=0
c     reject residual if limit is exceeded
      if(dabs(tres).gt.rejec) then
      irej=1
      write(*,202) tres
      tres=0.00d+00
      endif
      if (irej.eq.1) goto 10
      write(*,102) tres
  102 format(2x,'RESIDUAL ',e20.13)
  202 format(2x,'RESIDUAL ',e20.13,'  This residual rejected'
c     compute tqinvr and pinv
      do 4 i=1,6
      tqinvr(i)=tqinvr(i)+Tval(i)*tres*qinv
```

AD-A179 424 STAR SENSOR FINE ATTITUDE POINTING MODEL FOR A SPINNING 2/2
GEOSYNCHRONOUS SATELLITE(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. J V TAYLOR
UNCLASSIFIED DEC 86 AFIT/GA/AA/86D-14 F/G 22/1 NL



MICROCOPY RESOLUTION TEST CHART

```
      do 3 j=1,6
      pinv(i,j)=Tval(i)*Tval(j)*qinv+pinv(i,j)
    3 continue
    4 continue
c     now get another data line
      if(islit.ne.0) goto 10
   20 continue
      close(1)
c     if all residuals have been rejected at this point
c     need to widen rejection criterion and repeat routine
      tqmag=dsqrt(tqinvr(1)**2+tqinvr(2)**2+tqinvr(3)**2
     +      +tqinvr(4)**2+tqinvr(5)**2+tqinvr(6)**2)
      if(tqmag.eq.0.00d+00) rejec=rejec*1.00d+01
      if(tqmag.eq.0.00d+00) goto 30
c     at this point we have a successful residuals pass
c     compute covariance by inverting pinv
c     here an imsld routine is used
c     add -limsld to f77 statement to link it to the program
      do 300 j=1,6
      do 299 i=1,6
      tmat(i,j)=pinv(i,j)
  299 continue
  300 continue
c     call leqt2f(pinv,6,6,6,p,7,work,ier)
      call linv1f(tmat,6,6,p,0,work,ier)
c     call ppinv(p,pinv)
c     compute state corrections at epoch
      do 6 i=1,6
      delx(i)=0.00d+00
      do 5 j=1,6
      delx(i)=delx(i)+p(i,j)*tqinvr(j)
    5 continue
    6 continue
c     check for convergence
c     Dr Wiesels fast and dirty check
      do 7 i=1,6
c     write delx
c     write(*,*) delx(i)
      if(dabs(delx(i)).gt.0.10d+00*dsqrt(dabs(p(i,i)))) ifail=1
    7 continue
      if(ifail.eq.0) then
c     write out final p matrix
      write(*,103)
  103 format(2x,'p matrix')
      do 9 i=1,6
      write(*,104) (p(i,j),j=1,6)
    9 continue
  104 format(1x,6(1x,e12.5))
      endif
      return
      end
      include'debug'
```

PROGRAM LISTING   debug

```
      subroutine ppinv(p,pinv)
c     this subroutine multiplies p by pinv to see if it is identity
      double precision p(6,6),pinv(6,6),prod(6,6),sum
      do 2 i=1,6
      do 1 j=1,6
      sum=0.00d+00
      do 5 k=1,6
      sum=sum+p(i,k)*pinv(k,j)
    5 continue
      prod(i,j)=sum
    1 continue
    2 continue
      write(*,10)
   10 format(2x,'P MATRIX')
      write(*,*) ((p(i,j),j=1,6),i=1,6)
      write(*,11)
   11 format(2x,'PINVERSE MATRIX')
      write(*,*) ((pinv(i,j),j=1,6),i=1,6)
      write(*,12)
   12 format(2x,'PRODUCT P*PINV  (should be identity)')
      write(*,*) ((prod(i,j),j=1,6),i=1,6)
      return
      end
```

PROGRAM LISTING   dtdx0

```
      subroutine dtdxo(t0,x0)
c     this routine finds the partials of the estimate time
c     with respect to the initial state
      double precision t0,x0(6),xpass0(6),xdel,ttrue,tst1,tst2
      double precision sdat(2),Tval(6),xst(6)
      del=0.10d-02
c     open statfile for reading  and dtfile for writing
      open(1,file='statfile')
      open(2,file='dtfile')
   10 continue
c     get ttrue,tst,sdat(RA,dec),islit from statfile
      read(1,100) islit,ttrue
  100 format(7x,i1,45x,e20.13)
c     if islit=0 we have reached the end of the file
      if(islit.eq.0) goto 999
      read(1,101) (sdat(i),i=1,2)
  101 format(24x,2(2x,e20.13))
      read(1,102) tst1
  102 format(1x,/,5x,e20.13,/)
      do 2 idel=1,6
c     store original state in xpass0
      do 1 i=1,6
      xpass0(i)=x0(i)
    1 continue
c     vary the initial state component
      xdel=x0(idel)*del
      if(x0(idel).eq.0.00d+00) xdel=del*0.10d+00
      xpass0(idel)=xpass0(idel)-xdel
      call tstate(t0,tst1,xpass0,sdat,islit,xst,tst2)
      Tval(idel)=(tst2-tst1)/xdel
    2 continue
c     write data out to dtfile
      write(2,200) islit,ttrue,tst1,(Tval(i),i=1,6)
  200 format(2x,'Slit ',i1,2(2x,e20.13),/,
     + 3(2x,e20.13),/,3(2x,e20.13))
c     get another siting and repeat process
      if(islit.ne.0) goto 10
  999 continue
c     write out end of data flag
      write(2,201) islit,ttrue,tst1
  201 format(7x,i1,2(2x,e20.13),/,10x,'END OF DATA')
      close(1)
      close(2)
      return
      end
```

PROGRAM LISTING dyno1

```
      subroutine approx(t,t0)
      common /dyncom/ z(6),a(6)
c     this subroutine computes first order dynamics from:
c     t - time (from call)
c     t0- time at state 0 (from call)
c     z(i) - initial state (from common aprox)
c     outputs are:
c     a(i) - first order state approximation at time t
c     NOTE - states 1,2,3 are exact
c
      double precision ak,am,z,a,t,t0,ta,t1,d
c     define constants:
      ak=-1.00d+00
      am=7.2921152d-05
c     approximate solutions to equations of motion
      ta=t-t0
      a(1)=z(1)*dcos(z(3)*ak*ta)+z(2)*dsin(z(3)*ak*ta)
      a(2)=-z(1)*dsin(z(3)*ak*ta)+z(2)*dcos(z(3)*ak*ta)
      a(3)=z(3)
      d=1.0d+00/((ak-1.00d+00)*z(3))
      t1=(ak-1.00d+00)*z(3)*ta-z(6)
      a(6)=z(6)+z(3)*ta
      a(5)=z(5)+am*ta+d*(z(1)*(dcos(t1)-dcos(z(6)))
     +     +z(2)*(dsin(t1)+dsin(z(6))))
      a(4)=z(4)+d*(z(1)*(dsin(t1)+dsin(z(6)))
     +     -z(2)*(dcos(t1)-dcos(z(6))))
      return
      end
```

PROGRAM LISTING   dyno2

```
      subroutine approx2(t,t0)
      common /dyncom/ z(6),a(6)
c     this subroutine computes second order dynamics from:
c     t - time (from call)
c     t0- time at state 0 (from call)
c     z(i) - initial state (from common aprox)
c     outputs are:
c     a(i) - second order state approximations at time t
c     NOTE -  states 1,2,3 are exact
c
      double precision z,a,ak,am,t1,t0,ta,t,d,a6,a4
      double precision q1,q2,q3,q4,q5,q6,q7
      double precision r1,r2,r3,r4,r5,r6,r7
      double precision s1,s2,s3,s4,s5
c     define constants:
      ak=-1.00d+00
      am=7.2921152d-05
c     approximate solutions to equations of motion
      ta=t-t0
      a(1)=z(1)*dcos(z(3)*ak*ta)+z(2)*dsin(z(3)*ak*ta)
      a(2)=-z(1)*dsin(z(3)*ak*ta)+z(2)*dcos(z(3)*ak*ta)
      a(3)=z(3)
      d=1.0d+00/((ak-1.00d+00)*z(3))
      t1=(ak-1.00d+00)*z(3)*ta-z(6)
      a(6)=z(6)+z(3)*ta
      a6=z(4)+d*(z(1)*dsin(z(6))+z(2)*dcos(z(6)))
      a4=-z(4)*d*(z(1)*dcos(z(6))-z(2)*dsin(z(6)))-d*d*
     +   ((z(1)**2-z(2)**2)*dsin(z(6))*dcos(z(6))/2.00d+00+
     +   z(1)*z(2)*(dcos(z(6)))**2)
      s1=-(d*(z(1)**2+z(2)**2)/2.00d+00)*ta
      s2=(a6*d*z(1))*(dcos(t1)-dcos(z(6)))
      s3=(a6*d*z(2))*(dsin(t1)+dsin(z(6)))
      s4=(d*d*(z(1)**2-z(2)**2)/4.00d+00)*
     +   (dsin(2.00d+00*t1)+dsin(2.00d+00*z(6)))
      s5=(d*d*z(1)*z(2))*((dsin(t1))**2-(dsin(z(6)))**2)
      a(6)=a(6)+s1+s2+s3+s4+s5
c  end of a(6) calculation
c  start  a(5) calculation
      a(5)=z(5)+am*ta+d*(z(1)*(dcos(t1)-dcos(z(6)))
     +     +z(2)*(dsin(t1)+dsin(z(6))))
      r1=-(d*d*(z(1)**2+z(2)**2)/2.00d+00)
     +   *(z(1)*dsin(t1)-z(2)*dcos(t1))*ta
      r2=((a6*a6*d*z(2)/2.00d+00)+(a4*d*z(1))-(d**3*z(1)**2
     +   *z(2)/2.00d+00))*(dsin(t1)+dsin(z(6)))
      r3=((a6*a6*d*z(1)/2.00d+00)-(a4*d*z(2))-(d**3*3.00d+00
     +   *z(1)*z(2)**2/2.00d+00))*(dcos(t1)-dcos(z(6)))
      r4=(2.00d+00*d*a6*z(1)*z(2))*
     +   ((dsin(t1))**2-(dsin(z(6)))**2)
      r5=(a6*d*d*(z(1)**2-z(2)**2)/2.00d+00)*
     +   (dsin(2.00d+00*t1)+dsin(2.00d+00*z(6)))
```

```
      r6=d*d*d*(z(1)**2*z(2)-(z(2)**3/3.00d+00))*
     +   ((dsin(t1))**3+(dsin(z(6)))**3)
      r7=d*d*d*(z(1)*z(2)**2-(z(1)**3/3.00d+00))*
     +   ((dcos(t1))**3-(dcos(z(6)))**3)
      a(5)=a(5)+r1+r2+r3+r4+r5+r6+r7
c     end state 5 calculation
c     begin state 4 calculation
      a(4)=z(4)+d*(z(1)*(dsin(t1)+dsin(z(6)))
     +      -z(2)*(dcos(t1)-dcos(z(6))))
      q1=(d*d*(z(1)**2+z(2)**2)/2.00d+00)*
     +   (z(1)*dcos(t1)+z(2)*dsin(t1))*ta
      q2=-(d*a4*z(2)+(d*d*d*(z(1)**2+z(2)**2)*z(1)/2.00d+00))*
     +   (dsin(t1)+dsin(z(6)))
      q3=-(d*a4*z(1)+(d*d*d*(z(1)**2-z(2)**2)*z(2)/2.00d+00))*
     +   (dcos(t1)-dcos(z(6)))
      q4=(d*d*a6*(z(1)**2-z(2)**2)/2.00d+00)*
     +   ((dsin(t1))**2-(dsin(z(6)))**2)
      q5=-(d*d*a6*z(1)*z(2)/2.00d+00)*
     +   (dsin(2.00d+00*t1)+dsin(2.00d+00*z(6)))
      q6=d*d*d*(z(1)**3/6.00d+00-z(1)**2*z(2)/2.00d+00)*
     +   ((dsin(t1))**3+(dsin(z(6)))**3)
      q7=-(d*d*d*(z(1)**2+z(2)**2)*z(2)/6.00d+00)*
     +   ((dcos(t1))**3-(dcos(z(6)))**3)
      a(4)=a(4)+q1+q2+q3+q4+q5+q6+q7
      return
      end
```

PROGRAM LISTING   guessin

```
 6    5   30            no. odes,epoch updates,scans between update
 0.0000000000000d+00   start time
 0.1000000000000d+00   numerical integration timestep
 0.0000000000000d+00   initial sidereal time
 0.1000000000000d-01   initial true omega10
 0.5000000000000d-01   initial true omega20
 0.5223598776000d+00   initial true omega30
 0.5000000000000d-01   initial true psi10
 0.5000000000000d-01   initial true psi20
 0.8000000000000d+00   initial true psi30
10                     max number of residual passes
 0.0110000000000d+00   initial est. omega10
 0.0550000000000d+00   initial est. omega20
 5.7235987760000d-01   initial est. omega30
 0.0550000000000d+00   initial est. psi10
 0.0550000000000d+00   initial est. psi20
 0.8080000000000d+00   initial est. psi30
 1.0000000000000d-09   covariance
 0                     noise flag (0=no noise)
 1.0000000000000d+00   noise rnd seed
```

PROGRAM LISTING   haming

```
c     THIS NUMERICAL SUBROUTINE PROVIDED BY DR WILLIAM WIESEL
c
c
c
c
      subroutine haming(nxt)
c
c     haming is a fourth order predictor-corrector algorithm
c     for the integration of systems of ordinary differential equations
c     the common /ham/ contains most of the variables:
c        x is the independent variable, the 'time'
c        y contains 4 copies of the state vector, with
c               n odes being integrated
c        f contains the calculated equations of motion
c        errest is a truncation error estimate
c        n is the number of ode s
c        h is the integration timestep
c     nxt assumes the values 1,2,3,4,1,2,3,4...., and points to
c     the current value of the state vector
c
c     the user must supply a subroutine 'rhs(nxt)' which
c     calculates the equations of motion f(i,nxt) from the
c     state vector y(i,nxt)
c
c     to initialize haming, the initial conditions must be stored
c     in x(i,1), i=1,n ; x,n,and h must be initialized, and
c     then haming is called with nxt=0.  If haming returns with
c     nxt=1, initialization is successful.  If nxt=0 still, haming
c     did not initialize (h is usually too big)
c
      common /ham/ x,y(12,4),f(12,4),errest(12),n,h
      double precision x,y,f,errest,h,xo,tol,hh
      tol = 1.0d-12
c     branch on nxt: startup or propagating?
      if(nxt) 190,10,200
c
c     haming initialization: 4 point picard iteration
c
   10 xo = x
      hh = h/2.d+00
      call rhs(1)
      do 40 l = 2,4
      x = x + hh
      do 20 i = 1,n
   20 y(i,l) = y(i,l-1) + hh*f(i,l-1)
      call rhs(l)
      x = x + hh
      do 30 i = 1,n
   30 y(i,l) = y(i,l-1) + h*f(i,l)
   40 call rhs(l)
```

```
      jsw = -10
   50 isw = 1
      do 120 i = 1,n
      hh = y(i,1) + h*( 9.d+00*f(i,1) + 19.d+00*f(i,2) - 5.d+00*f(i,3)
     1      + f(i,4) ) / 24.d+00
      if( dabs( hh - y(i,2)) .lt. tol ) go to 70
      isw = 0
   70 y(i,2) = hh
      hh = y(i,1) + h*( f(i,1) + 4.d+00*f(i,2) + f(i,3))/3.d+00
      if( dabs( hh-y(i,3)) .lt. tol ) go to 90
      isw = 0
   90 y(i,3) = hh
      hh = y(i,1) + h*( 3.d+00*f(i,1) + 9.d+00*f(i,2) + 9.d+00*f(i,3)
     1      + 3.d+00*f(i,4) ) / 8.d+00
      if( dabs(hh-y(i,4)) .lt. tol ) go to 110
      isw = 0
  110 y(i,4) = hh
  120 continue
      x = xo
      do 130 l = 2,4
      x = x + h
  130 call rhs(l)
      if(isw) 140,140,150
  140 jsw = jsw + 1
      if(jsw) 50,280,280
  150 x = xo
      isw = 1
      jsw = 1
      do 160 i = 1,n
  160 errest(i) = 0.0
      nxt = 1
      go to 280
  190 jsw = 2
      nxt = iabs(nxt)
c
c     haming propagation section
c
  200 x = x + h
      np1 = mod(nxt,4) + 1
      go to (210,230),isw
  210 go to (270,270,270,220),nxt
  220 isw = 2
c     permute indices
  230 nm2 = mod(np1,4) + 1
      nm1 = mod(nm2,4) + 1
      npo = mod(nm1,4) + 1
c     predictor
      do 240 i = 1,n
      f(i,nm2) = y(i,np1) + 4.d+00*h*( 2.d+00*f(i,npo) - f(i,nm1)
     1      + 2.d+00*f(i,nm2) ) / 3.d+00
  240 y(i,np1) = f(i,nm2) - 0.925619835d+00*errest(i)
      call rhs(np1)
```

```
c     corrector
      do 250 i = 1,n
      y(i,np1) = ( 9.d+00*y(i,npo) - y(i,nm2) + 3.d+00*h*( f(i,np1)
     1     + 2.d+00*f(i,npo) - f(i,nm1) ) ) / 8.d+00
      errest(i) = f(i,nm2) - y(i,np1)
  250 y(i,np1) = y(i,np1) + 0.0743801653d+00 * errest(i)
      go to (260,270),jsw
  260 call rhs(np1)
  270 nxt = np1
  280 return
      end

      include 'thes1rhs'
```

PROGRAM LISTING matxmat

```
      subroutine matmat(id,rin1,rin2,rout)
c  This routine multiplies two square matrices together
      double precision rin1(3,3),rin2(3,3),rout(3,3),sum
      do 15 i=1,id
      do 10 j=1,id
      sum=0.00d+00
      do 5 k=1,id
      sum=sum+rin1(i,k)*rin2(k,j)
    5 continue
      rout(i,j)=sum
   10 continue
   15 continue
      return
      end
```

PROGRAM LISTING   noise

```
      subroutine nois(tseed,t)
c     this routine adds pseudo random noise to the
c     true slit crossing time
c     slit crossing time and noise to be added are
c     written to an output file
c     and noisy times are passed along
      common/cov/q
      double precision tseed,t,q,tnoise
c     pseudo random normal (0,1) information is
c     obtained through IMSL routine ggnqf
      rnoise= ggnqf(tseed)
      tnoise=rnoise
      tnoise=tnoise*dsqrt(q)
      write(*,101) t,tnoise
  101 format(5x,'TIME :',e20.13,2x,'NOISE :',e20.13)
      t=t+tnoise
      return
      end
```

PROGRAM LISTING   observ2

```
      subroutine obser(x,t,n,sdat,obs)
c      This routine computes the values of the observation
c      relations from a given state,time,and star data
      common/sidereal/tgo
      double precision x(6),t,tgo
      double precision p(3),romeg(3,3),r1(3,3),r2(3,3),r3(3,3)
      double precision rthet(3,3),obs(2,3),sdat(2)
      double precision tmat(3,3),tvec(3),romega,thetaz,thetai
      double precision tmat1(3,3),tvec1(3)
      double precision hr1(3,3),hr2(3,3),hr3(3,3),romdot(3,3),pro(3)
c     constants
c     tgo specifies the starting orbital position of the satellite
c     in order for the star data table to be valid in this case
c     tgo should be close to 0
c     thetai and thetaz are star sensor slit geometry parameters
c     for this simulation
c     thetai=15 degrees
c     thetaz=0.5 degrees
      thetai=2.617993878d-01
      thetaz=8.72664626d-03
c     romega is the satellite nominal orbital rate
c     for this simulation
c     romega=1 rev per 24 hrs
      romega=7.2921152d-05
c     nflag (n) from call tells what is desired from this routine
c     n=1    observation state desired
c     n=2    partial obs state wrt state is desired
c     n=3    partial obs state wrt t is desired
c
c     component setup
c     star vector
      p(1)=dcos(sdat(1))*dcos(sdat(2))
      p(2)=dsin(sdat(1))*dcos(sdat(2))
      p(3)=dsin(sdat(2))
c     romeg matrix
      romeg(1,1)=-dsin(romega*(t-tgo))
      romeg(1,2)=0.00d+00
      romeg(1,3)=-dcos(romega*(t-tgo))
      romeg(2,1)=-romeg(1,3)
      romeg(2,2)=0.00d+00
      romeg(2,3)=romeg(1,1)
      romeg(3,1)=0.00d+00
      romeg(3,2)=-1.00d+00
      romeg(3,3)=0.00d+00
c     r1 matrix
      r1(1,1)=dcos(x(5))
      r1(1,2)=0.00d+00
      r1(1,3)=dsin(x(5))
      r1(2,1)=0.00d+00
      r1(2,2)=1.00d+00
      r1(2,3)=0.00d+00
      r1(3,1)=-r1(1,3)
      r1(3,2)=0.00d+00
      r1(3,3)=r1(1,1)
c     r2 matrix
      r2(1,1)=1.00d+00
      r2(1,2)=0.00d+00
      r2(1,3)=0.00d+00
      r2(2,1)=0.00d+00
      r2(2,2)=dcos(x(4))
      r2(2,3)=-dsin(x(4))
      r2(3,1)=0.00d+00
      r2(3,2)=-r2(2,3)
      r2(3,3)=r2(2,2)
c     r3 matrix
      r3(1,1)=dcos(x(6))
      r3(1,2)=-dsin(x(6))
      r3(1,3)=0.00d+00
      r3(2,1)=-r3(1,2)
      r3(2,2)=r3(1,1)
      r3(2,3)=0.00d+00
      r3(3,1)=0.00d+00
      r3(3,2)=0.00d+00
      r3(3,3)=1.00d+00
c     rthet matrix
      rthet(1,1)=dcos(thetaz)
      rthet(1,2)=dcos(thetai)*dsin(thetaz)
      rthet(1,3)=-dsin(thetai)*dsin(thetaz)
      rthet(2,1)=-dsin(thetaz)
      rthet(2,2)=dcos(thetai)*dcos(thetaz)
      rthet(2,3)=-dsin(thetai)*dcos(thetaz)
      rthet(3,1)=0.00d+00
      rthet(3,2)=dsin(thetai)
      rthet(3,3)=dcos(thetai)
c     compute observation matrix
c     compute observation relation 1
      call matmat(3,r2,r3,tmat)
      call matmat(3,r1,tmat,tmat1)
      call matmat(3,romeg,tmat1,tmat)
      do 3 ik=1,3
      tvec(ik)=p(ik)
    3 continue
      call vecmat(3,tvec,tmat,tvec1)
      do 2 j=1,3
      obs(1,j)=tvec1(j)
    2 continue
c     compute observation relation 2
      call vecmat(3,tvec1,rthet,tvec)
      do 6 j=1,3
      obs(2,j)=tvec(j)
    6 continue
c     if n=1  G has been found so return
```

```
      if(n.eq.1) return
c     if n=2  compute H matrix
      if(n.eq.2) then
c        Since the left half of this matrix is zero
c        only the right half will be computed
c     compute h matrix
c     hr1 matrix
      hr1(1,1)=r1(3,1)
      hr1(1,2)=0.00d+00
      hr1(1,3)=r1(1,1)
      hr1(2,1)=0.00d+00
      hr1(2,2)=0.00d+00
      hr1(2,3)=0.00d+00
      hr1(3,1)=-hr1(1,3)
      hr1(3,2)=0.00d+00
      hr1(3,3)=hr1(1,1)
c     hr2 matrix
      hr2(1,1)=0.00d+00
      hr2(1,2)=0.00d+00
      hr2(1,3)=0.00d+00
      hr2(2,1)=0.00d+00
      hr2(2,2)=r2(2,3)
      hr2(2,3)=-r2(2,2)
      hr2(3,1)=0.00d+00
      hr2(3,2)=r2(2,2)
      hr2(3,3)=r2(2,3)
c     hr3 matrix
      hr3(1,1)=r3(1,2)
      hr3(1,2)=-r3(1,1)
      hr3(1,3)=0.00d+00
      hr3(2,1)=r3(1,1)
      hr3(2,2)=hr3(1,1)
      hr3(2,3)=0.00d+00
      hr3(3,1)=0.00d+00
      hr3(3,2)=0.00d+00
      hr3(3,3)=0.00d+00
c     compute h matrix
      call vecmat(3,p,romeg,tvec)
      do 13 i=1,3
      pro(i)=tvec(i)
   13 continue
      call matmat(3,r1,hr2,tmat)
      call matmat(3,tmat,r3,tmat1)
      call vecmat(3,pro,tmat1,tvec)
      obs(1,1)=tvec(2)
      call vecmat(3,tvec,rthet,tvec1)
      obs(2,1)=tvec1(2)
      call matmat(3,hr1,r2,tmat)
      call matmat(3,tmat,r3,tmat1)
      call vecmat(3,pro,tmat1,tvec)
      obs(1,2)=tvec(2)
      call vecmat(3,tvec,rthet,tvec1)
      obs(2,2)=tvec1(2)
      call matmat(3,r1,r2,tmat)
      call matmat(3,tmat,hr3,tmat1)
      call vecmat(3,pro,tmat1,tvec)
      obs(1,3)=tvec(2)
      call vecmat(3,tvec,rthet,tvec1)
      obs(2,3)=tvec1(2)
c     now Hmatrix has been computed
      endif
c     if n=2 we are done so return
      if(n.eq.2) return
c     if n=3  then compute gdot
c     gdot will be passed back as a matrix  but only row 2
c     is meaningful
      if(n.eq.3) then
c     romdot matrix
      romdot(1,1)=romega*romeg(1,3)
      romdot(1,2)=0.00d+00
      romdot(1,3)=romega*romeg(2,1)
      romdot(2,1)=romega*romeg(1,1)
      romdot(2,2)=0.00d+00
      romdot(2,3)=romdot(1,1)
      romdot(3,1)=0.00d+00
      romdot(3,2)=0.00d+00
      romdot(3,3)=0.00d+00
c     now get the gdot matrix
      call matmat(3,r2,r3,tmat)
      call matmat(3,r1,tmat,tmat1)
      call matmat(3,romdot,tmat1,tmat)
      call vecmat(3,p,tmat,tvec)
      do 21 i=1,3
      obs(1,i)=tvec(i)
   21 continue
      call vecmat(3,tvec,rthet,tvec1)
      do 22 i=1,3
      obs(2,i)=tvec1(i)
   22 continue
      endif
      return
      end


      include'matxmat'
      include'vecxmat'
```

PROGRAM LISTING ploterr

```
      subroutine plterr
c     this routine reads file (errlist) and
c     writes the information in plottable format in file (errplot)
      open(1,file='errlist')
      open(2,file='errplot')
c     scan errlist for min/max error, min/max t
      tmin=0.0
      tmax=0.0
      errmin=0.0
      errmax=0.0
   10 continue
      read(1,100) t,tperr
      if(t.ge.86399.0) goto 20
      if(t.lt.tmin) tmin=t
      if(t.gt.tmax) tmax=t
      if(tperr.lt.errmin) errmin=tperr
      if(tperr.gt.errmax) errmax=tperr
      if(t.lt.86399.0) goto 10
   20 continue
      close(1)
c     set scale factors
      tscale=tmax-tmin
      escale=errmax-errmin
      write(*,101) tscale,escale
c     read file again and write to errplot
      open(1,file='errlist')
   30 continue
      read(1,100) t,tperr
      if(t.ge.86399.0) goto 40
      tplot=(tmax-t)/tscale
      eplot=tperr/escale
      write(2,102) tplot,eplot
      if(t.lt.86399.0) goto 30
   40 continue
      close(1)
      close(2)
  100 format(2x,f15.7,2x,f15.7)
  101 format(2x,/,5x,'ERRPLOT PARAMETERS',/,
     + 5x,'Time scale factor :  ',f15.7,/,
     + 5x,'Error scale factor:  ',f15.7,/)
  102 format(2x,2f8.5,"\"+\"")
      return
      end
```

PROGRAM LISTING stardat

```
  1   0.1445190798496e+01  -0.5405672545077e-02   2.23  1852   36
  2   0.1400495825270e+01  -0.6927987503960e-02   4.73  1765   26
  3   0.1714109675279e+01  -0.2107969885740e-01   5.10  2395  107
  4   0.1463574933300e+01  -0.2113787649914e-01   1.70  1903   45
  5   0.1483420781311e+01  -0.3403392041833e-01   1.77  1948   50
  6   0.1412393152895e+01  -0.4207213125218e-01   3.36  1788   28
  7   0.1474657774023e+01  -0.4551915652532e-01   3.81  1931   48
  8   0.1689449627358e+01  -0.8293222830143e-01   5.05  2344  102
  9   0.1339918355738e+01  -0.8911360273638e-01   2.79  1666   10
 10   0.1460287896624e+01  -0.1033089473211e+00   2.76  1899   44
 11   0.1632311910943e+01  -0.1094175997039e+00   3.98  2227   87
 12   0.1382533478382e+01  -0.1197344348393e+00   3.60  1735   21
 13   0.1653554022376e+01  -0.1364120254716e+00   5.27  2273   93
 14   0.1410226035796e+01  -0.1365138363446e+00   4.14  1784   27
 15   0.1369174437342e+01  -0.1434466719854e+00   0.12  1713   19
 16   0.1345663397929e+01  -0.1531235530616e+00   4.27  1679   11
 17   0.1514160392769e+01  -0.1688509088789e+00   2.06  2004   57
 18   0.1673108982179e+01  -0.2010813224032e+00   5.22  2305   99
 19   0.1359495132309e+01  -0.2074663185843e+00   4.45  1696   13
 20   0.1804103214846e+01  -0.2097643354331e+00   4.07  2574  131
 21   0.1363603928118e+01  -0.2261801266776e+00   4.36  1705   17
 22   0.1391289213298e+01  -0.2302428653258e+00   4.29  1756   24
 23   0.1552026765444e+01  -0.2473131550399e+00   3.71  2085   67
 24   0.1510807906188e+01  -0.2587789985996e+00   3.55  1998   54
 25   0.1362374925464e+01  -0.2831505823522e+00   3.31  1702   14
 26   0.1589529527668e+01  -0.2876738939975e+00   4.93  2148   78
 27   0.1764811489972e+01  -0.2913681742481e+00  -1.46  2491  119
 28   0.1812720778029e+01  -0.2972877492952e+00   4.38  2596  133
 29   0.1448819628889e+01  -0.3112406870393e+00   2.58  1865   39
 30   0.1666862157924e+01  -0.3132429675426e+00   1.98  2294   96
 31   0.1733148308483e+01  -0.3180571673966e+00   4.43  2443  112
 32   0.1727905048521e+01  -0.3358401332220e+00   3.95  2429  110
 33   0.1429337391052e+01  -0.3625242782338e+00   2.84  1829   32
 34   0.1530021072289e+01  -0.3644198997272e+00   3.81  2035   62
 35   0.1329955434659e+01  -0.3907986121198e+00   3.19  1654    7
 36   0.1607062814476e+01  -0.3913706922636e+00   5.50  2180   80
 37   0.1500183214337e+01  -0.3918748984920e+00   3.60  1983   51
 38   0.1720923731484e+01  -0.4005870003427e+00   4.54  2414  108
 39   0.1706975641932e+01  -0.4085234003035e+00   4.34  2387  105
 40   0.1804183208993e+01  -0.4217394212523e+00   3.86  2580  132
 41   0.1823941790624e+01  -0.5052776666551e+00   1.50  2618  136
 42   0.1656833786818e+01  -0.5245732511658e+00   3.02  2282   94
 43   0.1785740896603e+01  -0.5670574740470e+00   3.96  2538  125
 44   0.1691194956471e+01  -0.5684488893120e+00   4.48  2361  103
 45   0.1664811395998e+01  -0.5834296320602e+00   3.85  2296   95
 46   0.1479544695916e+01  -0.5948421461150e+00   2.64  1956   49
 47   0.1382846183151e+01  -0.6092944419508e+00   4.83  1743   22
 48   0.1640609497089e+01  -0.6132214327683e+00   4.37  2256   90
 49   0.1557648180022e+01  -0.6158297303730e+00   4.36  2106   68
 50   0.1442783698610e+01  -0.6192670593725e+00   3.87  1862   35
 51   0.1528966602533e+01  -0.6243672992984e+00   3.12  2040   59
 52   0.1775108932588e+01  -0.6616931046119e+00   5.26  2518  122
 53   0.1565000379386e+01  -0.7472724156126e+00   3.96  2120   73
 54   0.1733490102183e+01  -0.7536671080672e+00   3.17  2451  113
 55   0.1438725808154e+01  -0.8218464560506e+00   5.46  1856   33
 56   0.1328908236997e+01  -0.8656542202814e+00   5.03  1663    6
 57   0.1787006260213e+01  -0.8830638795723e+00   2.93  2553  127
 58   0.1391841901061e+01  -0.8835244525694e+00   5.45  1767   25
 59   0.1513709516042e+01  -0.8913687379308e+00   3.85  2020   56
 60   0.1529504745830e+01  -0.9095298584275e+00   5.17  2049   61
 61   0.1673807113907e+01  -0.9195606534910e+00  -0.72  2326  100
 62   0.1721912751422e+01  -0.9243700052082e+00   4.39  2435  109
 63   0.1786860816248e+01  -0.9355595049697e+00   4.40  2554  126
 64   0.1614415013840e+01  -0.9593153753471e+00   4.81  2212   82
 65   0.1525177783726e+01  -0.9803562891100e+00   4.51  2042   58
 66   0.1331875296809e+01  -0.1003452812881e+01   4.72  1674    8
 67   0.1780381281371e+01  -0.1080790291302e+01   3.27  2550  124
 68   0.1455124630781e+01  -0.1090830782639e+01   3.40  1922   41
 69   0.1504233832699e+01  -0.1147403691095e+01   4.35  2015   53
 70   0.1369065354264e+01  -0.1172909738861e+01   4.83  1744   18
 71   0.1906081348557e+01  -0.1185587616624e+01   3.98  2803  282
 72   0.1609360831213e+01  -0.1201489505367e+01   5.06  2221   81
 73   0.1871167491418e+01  -0.1229972309135e+01   3.62  2735  281
 74   0.1796074700175e+01  -0.1238214141715e+01   5.40  2602  128
 75   0.2015935280576e+01  -0.1266570893927e+01   3.95  3024  283
 76   0.9925251125630e+00  -0.1296546923834e+01   3.24  1208  279
 77   0.1617505701057e+01  -0.1304604527215e+01   5.09  2261   83
 78   0.1289907400470e+01  -0.1308318200013e+01   5.47  1629  280
 79   0.1450775852117e+01  -0.1332607365440e+01   5.19  1953   40
 80   0.2177058259431e+01  -0.1341653988730e+01   4.07  3318  299
 81   0.2186584848196e+01  -0.1351495706458e+01   4.35  3340  300
 82   0.2277138347691e+01  -0.1377195679697e+01   5.47  3502  301
 83   0.6631305772321e+00  -0.1381898372405e+01   5.28   776  278
 84   0.2817157762683e+01  -0.1404262827517e+01   4.45  4234  285
 85   0.2352492938072e+01  -0.1494050321270e+01   5.42  3678  284
 86   0.5472610770051e+01  -0.1553658163371e+01   5.47  7228  294
 87   0.4912760051305e+01  -0.1529330212849e+01   5.28  6721  249
 88   0.6135370464754e+01  -0.1528346041077e+01   5.49  8862  298
 89   0.3373881433046e+01  -0.1484218299816e+01   5.46  4870  286
 90   0.5953914400213e+01  -0.1421803386502e+01   4.15  8630  297
 91   0.5839660784012e+01  -0.1405295480658e+01   5.10  8481  296
 92   0.3865409783773e+01  -0.1378465891542e+01   3.83  5470  287
 93   0.4324242299709e+01  -0.1376444218491e+01   3.89  6102  289
 94   0.4267308205123e+01  -0.1372856597251e+01   4.68  6020  288
 95   0.4366988322096e+01  -0.1352407156179e+01   4.24  6163  290
 96   0.5510346242966e+01  -0.1345372509665e+01   5.15  8021  295
 97   0.5230839035428e+01  -0.1273271019001e+01   3.96  7590  293
 98   0.4892303338083e+01  -0.1246921395429e+01   4.01  6982  242
 99   0.4539448668693e+01  -0.1223626098049e+01   5.41  6411  161
100   0.4393939114522e+01  -0.1204291728444e+01   1.92  6217  291
101   0.4953935277302e+01  -0.1173811492308e+01   3.90  7107  262
102   0.4643499380960e+01  -0.1129538306944e+01   3.62  6582  192
```

103	0.4743317669804e+01	-0.1111260831164e+01	4.35	6745	216
104	0.4933980346074e+01	-0.1085715998303e+01	4.22	7074	256
105	0.4807509425246e+01	-0.1073421123348e+01	4.36	6855	229
106	0.4580173018050e+01	-0.1058930042418e+01	3.62	6500	169
107	0.4400156849983e+01	-0.1030006058199e+01	3.76	6229	292
108	0.4555687502957e+01	-0.9837451367414e+00	3.34	6482	164
109	0.4781416752981e+01	-0.9778982837465e+00	5.33	6819	221
110	0.4555352981533e+01	-0.9689437750552e+00	2.85	6461	163
111	0.4962080147146e+01	-0.9243263719769e+00	5.03	7134	265
112	0.4637834333054e+01	-0.9045508219218e+00	5.15	6585	191
113	0.4736052736750e+01	-0.8743032963535e+00	3.66	6743	212
114	0.4584296358186e+01	-0.8703132797574e+00	2.95	6510	173
115	0.4832983960068e+01	-0.8566172932643e+00	4.13	6905	235
116	0.4825071800847e+01	-0.8024684532141e+00	3.51	6897	233
117	0.4756364005937e+01	-0.8021193873637e+00	4.53	6783	218
118	0.4920621305312e+01	-0.7626749462634e+00	5.49	7062	254
119	0.4498767952733e+01	-0.7543264546736e+00	3.33	6380	153
120	0.4608563707122e+01	-0.7503025011199e+00	1.87	6553	180
121	0.4853746105964e+01	-0.7387057578662e+00	4.64	6951	238
122	0.4653484118737e+01	-0.7002648810860e+00	3.03	6615	194
123	0.4631296620591e+01	-0.6810808037220e+00	2.41	6580	186
124	0.4605393025689e+01	-0.6741431199445e+00	4.29	6546	179
125	0.4580216651170e+01	-0.6507363154174e+00	2.69	6508	170
126	0.4592637577626e+01	-0.6474056454278e+00	1.63	6527	177
127	0.4663534306404e+01	-0.6464602587495e+00	3.21	6630	197
128	0.4784725606355e+01	-0.6417139328108e+00	3.11	6832	222
129	0.4813370822679e+01	-0.6002720593442e+00	1.85	6879	231
130	0.4733383837519e+01	-0.5310212731254e+00	2.99	6746	211
131	0.4565621335243e+01	-0.5210535038405e+00	4.29	6492	168
132	0.4799662715816e+01	-0.5207335268109e+00	2.70	6859	224
133	0.4729973173213e+01	-0.5162974816282e+00	4.30	6742	209
134	0.4653855001093e+01	-0.4856524088413e+00	4.20	6616	195
135	0.4907378619444e+01	-0.4713746459307e+00	3.17	7039	246
136	0.4949339243549e+01	-0.4593173296800e+00	2.02	7121	259
137	0.4830264155370e+01	-0.4438566213874e+00	2.81	6913	234
138	0.4542466633893e+01	-0.4360705136677e+00	3.27	6453	162
139	0.4561519811528e+01	-0.4217054842946e+00	4.17	6486	165
140	0.4636030826215e+01	-0.3783292042400e+00	4.87	6595	188
141	0.4960247551376e+01	-0.3687541340369e+00	3.51	7150	264
142	0.4538176032814e+01	-0.3682159908508e+00	4.39	6445	159
143	0.4768399505641e+01	-0.3676342144334e+00	3.86	6812	219
144	0.4491990257415e+01	-0.2741330478824e+00	2.43	6378	152
145	0.4610716279964e+01	-0.2686013237802e+00	3.54	6561	181
146	0.4835929203181e+01	-0.2544108273323e+00	4.70	6930	236
147	0.4627493257179e+01	-0.2245899378033e+00	4.26	6581	185
148	0.4537659706181e+01	-0.2239596800178e+00	4.33	6446	158
149	0.4704418644054e+01	-0.1705671493102e+00	3.34	6698	205
150	0.4893139641683e+01	-0.1582722743557e+00	4.72	7020	243
151	0.4898601067691e+01	-0.1447120356933e+00	4.90	7032	244
152	0.4862327308121e+01	-0.1440914741814e+00	3.85	6973	239
153	0.4915959821601e+01	-0.9987646645842e-01	5.20	7066	252
154	0.4563199690961e+01	-0.8855606700303e-01	4.54	6493	167

```
155   0.4914636280334e+01  -0.8316978700520e-01   4.22  7063 251
156   0.4451644062923e+01  -0.7330867673015e-01   4.82  6318 148
157   0.4801866193970e+01  -0.5067757409300e-01   3.26  6869 226
158   0.4838387208489e+01  -0.3483871112908e-01   5.39  6935 237
159   0.4716606860026e+01   0.2277169860469e-01   4.45  6723 208
160   0.4732772972169e+01   0.4367201640005e-01   4.03  6752 210
161   0.4656167562470e+01   0.4733720782972e-01   3.75  6629 196
162   0.4711814476677e+01   0.5116238777417e-01   3.97  6714 207
163   0.4800062687096e+01   0.5880789952627e-01   4.86  6866 225
164   0.4562923347218e+01   0.7248449347215e-01   4.34  6498 166
165   0.4954327976356e+01   0.7300324411101e-01   4.06  7141 263
166   0.4636932579773e+01   0.7981487633148e-01   2.77  6603 189
167   0.4868261427745e+01   0.1589897986038e+00   5.39  6985 240
168   0.4741252363620e+01   0.1668680209229e+00   3.73  6771 213
169   0.4599880694051e+01   0.2193975832780e+00   2.08  6556 178
170   0.4470915406750e+01   0.2227282532676e+00   4.91  6355 150
171   0.4511414317718e+01   0.2514534638771e+00   3.08  6406 154
172   0.4969468707536e+01   0.2626090266809e+00   4.02  7176 267
173   0.4709625543017e+01   0.2923620422945e+00   4.67  6713 206
174   0.4536248898536e+01   0.3154197809710e+00   5.00  6452 157
175   0.4914570830515e+01   0.3170148179821e+00   4.36  7069 250
176   0.4908716705315e+01   0.3583306398917e+00   4.19  7061 247
177   0.4747710081728e+01   0.3632272580715e+00   4.36  6787 217
178   0.4812905401521e+01   0.3798175822412e+00   3.84  6895 230
179   0.4637005301825e+01   0.4247064809810e+00   5.71  6608 190
180   0.4513399629686e+01   0.4338355225975e+00   3.14  6410 155
181   0.4689670611789e+01   0.4546922071616e+00   5.46  6685 200
182   0.4581976524819e+01   0.4559090895013e+00   4.41  6526 172
183   0.4910702017284e+01   0.4650429792546e+00   4.83  7064 248
184   0.4650655230797e+01   0.4839652572308e+00   3.42  6623 193
185   0.4742663171265e+01   0.5019518448023e+00   3.83  6779 214
186   0.4700004415501e+01   0.5104894137278e+00   3.70  6703 203
187   0.4703269635546e+01   0.5269149012459e+00   4.41  6707 204
188   0.4538197849409e+01   0.5670041445421e+00   5.39  6458 160
189   0.4967047063254e+01   0.5701602816065e+00   3.24  7178 266
190   0.4928402564756e+01   0.5819654947431e+00   3.45  7106 255
191   0.4455098360402e+01   0.5862560958214e+00   5.25  6332 149
192   0.4796681111705e+01   0.6293123988464e+00   4.33  6872 223
193   0.4513886867353e+01   0.6427368896781e+00   3.16  6418 156
194   0.4947841169385e+01   0.6436531875355e+00   4.30  7139 258
195   0.4693721230151e+01   0.6501690834105e+00   3.86  6695 201
196   0.4905415124104e+01   0.6560401770895e+00   4.36  7056 245
197   0.4871272120538e+01   0.6766399104025e+00   0.03  7001 241
198   0.4742983148350e+01   0.7585103967422e+00   5.00  6791 215
199   0.4951775432297e+01   0.7666407221754e+00   4.04  7157 261
200   0.4620875550544e+01   0.8030987109996e+00   3.80  6588 184
201   0.4696004702631e+01   0.8986748801060e+00   2.23  6705 202
202   0.4581845625146e+01   0.9130253650688e+00   2.79  6536 171
203   0.4590041400308e+01   0.9631260108812e+00   4.87  6555 175
204   0.4589648701393e+01   0.9633247844904e+00   4.88  6554 174
205   0.4682980183128e+01   0.9926511640646e+00   3.75  6688 199
206   0.4815727017142e+01   0.1026103308066e+01   4.98  6923 232
```

207	0.4934802105486e+01	0.1036187432634e+01	4.66	7125	257
208	0.4772631929134e+01	0.1123851442464e+01	5.03	6850	220
209	0.4488725037439e+01	0.1147267943264e+01	3.17	6396	151
210	0.5028962618588e+01	0.1180443743467e+01	3.07	7310	271
211	0.4590325016423e+01	0.1189354618927e+01	5.05	6566	176
212	0.4612207081929e+01	0.1200180508427e+01	4.80	6596	182
213	0.4310999614015e+01	0.1200815614350e+01	5.00	6161	275
214	0.5115530949470e+01	0.1215355176648e+01	4.68	7462	272
215	0.5184631443434e+01	0.1225720493152e+01	3.83	7582	273
216	0.4950590062902e+01	0.1244012513342e+01	4.82	7180	260
217	0.4803931500224e+01	0.1244943355610e+01	4.22	6920	227
218	0.4634772734699e+01	0.1259371410762e+01	4.58	6636	187
219	0.4805487752251e+01	0.1269314939363e+01	3.57	6927	228
220	0.5043368857013e+01	0.1279796611150e+01	4.45	7352	270
221	0.5374406910889e+01	0.1307280698735e+01	5.20	7879	269
222	0.4916927024839e+01	0.1316264296247e+01	5.35	7124	253
223	0.4267090038967e+01	0.1322809280943e+01	4.95	6116	274
224	0.4669373887138e+01	0.1343302355246e+01	5.04	6701	198
225	0.3703988644503e+01	0.1354729413712e+01	4.82	5321	277
226	0.5277141165965e+01	0.1355514811875e+01	4.39	7750	268
227	0.4121544123631e+01	0.1358612771298e+01	4.32	5903	276
228	0.4396128048320e+01	0.1432294754563e+01	4.23	6322	142
229	0.5997729436649e+01	0.1470672605564e+01	4.71	8748	144
230	0.5823356499997e+01	0.1501516451960e+01	5.27	8546	143
231	0.4612810674942e+01	0.1511382410372e+01	4.36	6789	183
232	0.5930192465628e+00	0.1556731882109e+01	2.02	424	138
233	0.1979428810271e+01	0.1519410924932e+01	5.07	2609	139
234	0.2897973779211e+00	0.1504027786829e+01	4.25	285	137
235	0.2745737435213e+01	0.1442311005216e+01	5.26	4084	141
236	0.2508736267033e+01	0.1420634985530e+01	4.29	3751	140
237	0.1761080848613e+01	0.1389005740971e+01	5.45	2401	117
238	0.1823090942558e+01	0.1343893827937e+01	4.55	2527	135
239	0.9979065444240e+00	0.1244177349994e+01	4.63	1148	145
240	0.1645583685430e+01	0.1209983441061e+01	4.80	2209	91
241	0.1797856390505e+01	0.1202672450749e+01	5.12	2511	130
242	0.1276279287920e+01	0.1157468423116e+01	4.29	1542	146
243	0.1620065517349e+01	0.1147083714065e+01	5.32	2165	85
244	0.1317876301730e+01	0.1054547326740e+01	4.03	1603	3
245	0.1650456062926e+01	0.1030059387704e+01	4.48	2238	92
246	0.1814866078568e+01	0.1020047985188e+01	4.35	2560	134
247	0.1542638348342e+01	0.9722308118136e+00	4.99	2029	65
248	0.1563160511522e+01	0.9474471364320e+00	3.72	2077	70
249	0.1291718179577e+01	0.9377411665349e+00	4.47	1568	147
250	0.1504030211008e+01	0.8695375778675e+00	5.47	1971	52
251	0.1674228901754e+01	0.8603988399774e+00	4.91	2289	101
252	0.1773596313958e+01	0.8518467266415e+00	5.22	2487	121
253	0.1376810252821e+01	0.8025605678135e+00	0.08	1708	20
254	0.1565494889285e+01	0.8017460808291e+00	4.26	2091	74
255	0.1563778648965e+01	0.7844721693690e+00	1.90	2088	71
256	0.1312720308294e+01	0.7644784531574e+00	2.99	1605	1
257	0.1769865672737e+01	0.7608568949590e+00	5.25	2483	120
258	0.1332667967150e+01	0.7193326031666e+00	3.17	1641	9

```
259   0.1315069230523e+01   0.7165303800894e+00   3.75  1612    2
260   0.1387747649475e+01   0.6996394714373e+00   4.71  1729   23
261   0.1528973874766e+01   0.6832091357824e+00   3.97  2012   60
262   0.1362949429648e+01   0.6713796819618e+00   4.86  1689   15
263   0.1564964018360e+01   0.6494806479832e+00   2.62  2095   72
264   0.1796678293153e+01   0.5930774243155e+00   3.60  2540  129
265   0.1447394276552e+01   0.5616760421860e+00   4.76  1843   38
266   0.1633584546821e+01   0.5149545477314e+00   4.35  2219   88
267   0.1419439919862e+01   0.4990817478098e+00   1.65  1791   30
268   0.1537431449517e+01   0.4818757102649e+00   4.58  2034   63
269   0.1758324683002e+01   0.4389066737026e+00   2.98  2473  115
270   0.1584664422336e+01   0.4060556986663e+00   4.16  2134   77
271   0.1666891246744e+01   0.3930917808317e+00   2.88  2286   97
272   0.1631628323645e+01   0.3929123997697e+00   3.28  2216   86
273   0.1318458078182e+01   0.3764432790203e+00   4.64  1602    4
274   0.1469210892204e+01   0.3688607930467e+00   3.00  1910   47
275   0.1542274738081e+01   0.3538509614776e+00   4.41  2047   64
276   0.1693158451984e+01   0.3529540561675e+00   4.15  2343  104
277   0.1583879024297e+01   0.3515044632607e+00   4.63  2135   76
278   0.1445583497619e+01   0.3243452008414e+00   4.38  1845   37
279   0.1423592348999e+01   0.3132817526371e+00   5.42  1808   31
280   0.1512029636609e+01   0.3093450655460e+00   5.49  1990   55
281   0.1731439340257e+01   0.2864667079314e+00   1.93  2421  111
282   0.1325061240603e+01   0.2684898166335e+00   4.68  1638    5
283   0.1599972414388e+01   0.2578045231005e+00   4.42  2159   79
284   0.1619047408619e+01   0.2480646162457e+00   4.48  2199   84
285   0.1758921003781e+01   0.2311591631832e+00   4.49  2478  116
286   0.1764615140515e+01   0.2253753359668e+00   3.36  2484  118
287   0.1638747812664e+01   0.2142827989416e+00   5.04  2241   89
288   0.1458586200493e+01   0.1732190801462e+00   3.39  1879   43
289   0.1745874667614e+01   0.1729766733057e+00   4.65  2456  114
290   0.1345968830465e+01   0.1712264959166e+00   5.43  1672   12
291   0.1577472211334e+01   0.1683951840186e+00   4.12  2124   75
292   0.1457211753595e+01   0.1654572131107e+00   4.41  1876   42
293   0.1466309282437e+01   0.1620198841112e+00   4.09  1907   46
294   0.1546063557027e+01   0.1292416311271e+00   0.50  2061   66
295   0.1710706283188e+01   0.1281992817125e+00   4.50  2385  106
296   0.1415018419006e+01   0.1105908488123e+00   1.64  1790   29
297   0.1439700283487e+01   0.1036240762139e+00   4.20  1839   34
298   0.1670920048380e+01   0.8031423442309e-01   4.33  2298   98
299   0.1363451211781e+01   0.4963037654166e-01   4.46  1698   16
300   0.1776097952491e+01   0.4241150082900e-01   4.47  2506  123
301   0.1562193308231e+01   0.9638095981716e-02   5.22  2103   69
999   0.             e+00   0.             e+00   0.       0    0
```

PROGRAM LISTING   theslrhs

```
      subroutine rhs(nxt)
      common /ham/ t,x(12,4),f(12,4),err(12),n,h
      double precision t,x,f,err,h,ak,aomega
c     it is the job of rhs to calculate the current equations
c     of motion f(i,nxt), i=1,n from the current value of
c     the state vector x(i,nxt), i = 1,n, at the current time
c     t.  The other three copies of the state vector are ignored.
      ak=-1.00d+00
      aomega=7.2921152d-05
      f(1,nxt)=ak*x(2,nxt)*x(3,nxt)
      f(2,nxt)=-ak*x(1,nxt)*x(3,nxt)
      f(3,nxt)=0.00d+00
      f(4,nxt)=x(1,nxt)*dcos(x(6,nxt))-x(2,nxt)*dsin(x(6,nxt))
      f(5,nxt)=x(1,nxt)*dsin(x(6,nxt))/dcos(x(4,nxt))+aomega
     +         +x(2,nxt)*dcos(x(6,nxt))/dcos(x(4,nxt))
      f(6,nxt)=x(1,nxt)*dsin(x(6,nxt))*dtan(x(4,nxt))+x(3,nxt)
     +         +x(2,nxt)*dcos(x(6,nxt))*dtan(x(4,nxt))
      return
      end
```

PROGRAM LISTING   torder

```
      subroutine order(inoise,tseed)
c     this routine reads the data file 'sitings' which was created by
c     'truth' and organizes it in time order.
c     the routine uses a modified bubble sorting technique
c     if noise is to be added to the system, it is done here
c
      double precision tslit(100),sdat(2,100)
      double precision tper,tmin,tmin2,zero,tseed
      dimension islit(100),id1(100),id2(100),iseq(100)
c     constants
      tper=12.00d+00*2.00d+00
c     open files and read data
      open(1,file='sitings')
      open(2,file='stars')
      if(inoise.ne.0) write(*,102)
  102 format(2x,//,15x,'NOISE ADDED TO TRUE SLIT CROSSING TIME')
      k=0
    1 continue
      k=k+1
      read(1,101) islit(k),id1(k),id2(k),tslit(k),
     +   (sdat(ij,k),ij=1,2)
  101 format(6x,i1,9x,i3,4x,i4,15x,e20.13,/,
     +   10x,e20.13,10x,e20.13,/)
c     if inoise not equal 0  noise will be added
      if(inoise.ne.0) then
      call nois(tseed,tslit(k))
    7 continue
      endif
      if(id1(k).ne.999) goto 1
      close (1)
c     end of data reading
c     now data will be sorted by time
      tmin2=tslit(1)+tper
      do 3 i=1,k-1
      jmin=i
      tmin=tslit(1)
      imin=1
      do 2 j=1,k-1
      if(tslit(j).lt.tmin.and.tslit(j).lt.tmin2) then
      tmin=tslit(j)
      imin=j
      endif
    2 continue
      iseq(imin)=jmin
      tslit(imin)=tslit(imin)+2.0d+00*tper
    3 continue
c     now convert data back to original data and write to 'stars'
      do 4 j=1,k-1
      tslit(j)=tslit(j)-2.00d+00*tper
    4 continue
```

```
      do 6 j=1,k-1
      do 5 i=1,k-1
      if(iseq(i).eq.j) then
      write(2,101) islit(i),id1(i),id2(i),tslit(i),
     +  (sdat(ij,i),ij=1,2)
      endif
    5 continue
    6 continue
      zero=0.00d+00
      izero=0
      iflag=999
      write(2,101) izero,iflag,izero,zero,zero,zero
      return
      end


      include'noise'
```

PROGRAM LISTING   tresid

```
      subroutine resid(t,x0)
c     This routine reads in initial data
c     reads the stars file (actual sitings)
c     and gets an estimated slit crossing time from tstate
      double precision x0(6),t0,tflag
      double precision xst(6),tst,ttrue,sdat(2)
c     read star crossing info
      open (1,file='stars')
   10 continue
      read(1,100)islit,id1,id2,ttrue,sdat(1),sdat(2)
  100 format(6x,i1,9x,i3,4x,i4,15x,e20.13,/,
     + 10x,e20.13,10x,e20.13,/)
      if(id1.eq.999) goto 999
      call tstate(t0,ttrue,x0,sdat,islit,xst,tst)
c     write out true and estimated conditions to statfile
      open(2,file='statfile')
      write(2,101) islit,id1,id2,ttrue
  101 format(2x,'Slit ',i1,' of star ',i4,'(id-',i4,') crossed at ',
     + 'true time :',e20.13)
      write(2,102) sdat(1),sdat(2)
  102 format(5x,'star data (RA,Dec):',2(2x,e20.13))
      write(2,104) tst
  104 format(2x,'ESTIMATION   TIME        ',/,5x,e20.13,/)
      if(id1.ne.999) goto 10
  999 continue
c     write end of data flag
      iflag=0
      tflag=0.00d+00
      write(2,101) iflag,iflag,iflag,tflag
      close(2)
      close(1)
      return
      end
```

PROGRAM LISTING truth

```
      subroutine truth(t0,tend,h,x0,tfin,xout)
c     This routine is called by program main
c     truth inputs are initial time (t0), integration timestep (h)
c           and initial state (z)
c     truth outputs are final time (t) and state (xout)
c     This version of truth records star sightings for a 12 second
c     period from t0. Each star is analyzed over this period, and
c     outputs are sent to 'sitings'. 'sitings' stores data as
c     id1,id2,time,star r.a.,star dec.,and true state at t
c     although true state is stored, the estimation algorithm
c     will not access this state
c     true state will be used only for result analysis
c
      common/ham/hamt,x(12,4),f(12,4),errest(12),n,hamh
      double precision t0,z(6),t,xout(6),hamt,hamh
      double precision x,f,errest,h
      double precision sdat(2),sold(2),obs(2,3)
      double precision alitx,dalit,thetai,fnterp,fov
      double precision tper,x0(6),xflag(3),xhold(2,3),tfin,tend
      dimension ihold(2,3)
c     constants
      thetai=2.61799388d-01
      fov=2.617993878d-02
      tper=tend
      nflag=1
c     open star data table for reading
      open(1,file='stardat')
      open(2,file='sitings')
c     read stars
      do 9 i=1,6
      z(i)=x0(i)
    9 continue
   11 continue
      read (1,101) id1,(sdat(i),i=1,2),id2
  101 format(1x,i3,2(2x,e20.13),9x,i4)
c     write(*,*) id1,id2
c     end of star reading
c     setup data for haming calls
      n=6
      t=t0
      hamt=t-t0
      hamh=h
      do 1 i=1,n
      x(i,1)=x0(i)
    1 continue
c     initialize haming
c     write(*,*)id2,t0,t,hamt
      nxt=0
      call haming(nxt)
      if(nxt.ne.0) goto 10
      write(*,2)
    2 format(2x,'haming did not initialize')
      stop
   10 continue
      if(id1.eq.999) goto 999
c     make first call to obser
      t=hamt
      ifov=0
      imiss=0
      call obser(z,t,nflag,sdat,obs)
   20 continue
c     store row 2 of obs in sold
      do 3 j=1,2
      sold(j)=obs(j,2)
    3 continue
c     integrate state equations
      call haming(nxt)
c     write(*,*) id1,id2,t0,t,hamt
c     now call obser at new time
      do 4 i=1,6
      z(i)=x(i,nxt)
    4 continue
      t=hamt
      call obser(z,t,nflag,sdat,obs)
c     check for slit crossings
      do 7 islit=1,2
      slitx=sold(islit)*obs(islit,2)
        if(slitx.le.0.00d+00) then
c        now check to see if crossing is forward looking
         if(obs(islit,1).gt.0.00d+00) then
c           see if this is the first slit crossed
c            and advance crossing counter imiss
             imiss=imiss+1
            if(ifov.eq.0) then
c              see if the slit is in the sensor fov
               dslit=dcos(fov)
               if(islit.eq.2) dslit=dcos(fov/dcos(thetai))
                  if(obs(islit,1).ge.dslit) ifov=1
c     write(*,*) islit,id1,id2,t,obs(islit,1),dslit,slitx
            endif
c
c           now we have a confirmed siting if ifov.gt.0
c
            if(ifov.gt.0) then
c           hold this siting
            ihold(islit,1)=islit
            ihold(islit,2)=id1
            ihold(islit,3)=id2
            xhold(islit,2)=sdat(1)
            xhold(islit,3)=sdat(2)
c           need to interpolate other hold params based on obs(x,2)
            k=nxt-1
```

```
               if(nxt.eq.1) k=4
            fnterp=-sold(islit)/(obs(islit,2)-sold(islit))
            xhold(islit,1)=(t-h)+h*fnterp
            ifov=ifov+1
            endif
         endif
        endif
    7 continue
c
c     now if ifov=3 want to write star to sitings
      if(ifov.eq.3) then
         do 8 iw=1,2
         write(2,102)(ihold(iw,iww),iww=1,3),(xhold(iw,iww),iww=1,3)
  102 format(1x,'Slit ',i1,' of star ',i3,'(id-',i4,
     + ') crossed at t=',e20.13,/,
     +      5x,'RA = ',e20.13,5x,'DEC= ',e20.13,/)
    8    continue
      endif
c
c     if we havent confirmed this star  continue integrating to tper
      if(t.le.tper.and.ifov.ne.3.and.imiss.lt.2) goto 20
c     if we havent reached the end of the starfile, get another star
      if(id1.ne.999) goto 11
c     TEMP END OF PROGRAM
      tfin=hamt+t0
      do 111 iend=1,6
      xout(iend)=x(iend,nxt)
  111 continue
  999 continue
c     now write data flag at end of file
      do 998 l=1,3
      xflag(l)=0.00d+00
  998 continue
      izero=0
      iflag=999
      write(2,102) izero,iflag,izero,(xflag(l),l=1,3)
      close(1)
      close(2)
c     now compute final true state to pass back to main program
  997 continue
      call haming(nxt)
      if(hamt.lt.tend) goto 997
      tfin=hamt+t0
      do 996 i=1,6
      xout(i)=x(i,nxt)
  996 continue
      return
      end


      include'haming'
      include'observ2'
```

PROGRAM LISTING tstate

```
      subroutine tstate(tepoch,t,x0,sdat,islit,xst,tst)
c     This routine uses a Newton-Rhaphson method to estimate a slit
c     crossing time
      common/dyncom/z(6),a(6)
      common/ham/tham,xham(12,4),f(12,4),errest(12),nham,hham
      double precision xst(6),x0(6),t,sdat(2),tst,told,tnew,tol
      double precision tdif,h(6),a,z,obs(2,3),dgdt
      double precision xham,f,errest,hham,tham
c     constants
      nham=6
      nxt=1
      tol=1.00d-08
      maxit=5
      do 1 i=1,6
      z(i)=x0(i)
    1 continue
      iter=0
      told=t
      t0=tepoch
    2 continue
      iter=iter+1
c     get state from dyno (1 or 2, depending on order)
      call approx2(told,t0)
c     get h from obser
      call obser(a,told,2,sdat,obs)
c     recall left side of h=0
      h(1)=0.00d+00
      h(2)=0.00d+00
      h(3)=0.00d+00
      h(4)=obs(islit,1)
      h(5)=obs(islit,2)
      h(6)=obs(islit,3)
c     get xdot from rhs
      do 3 i=1,6
      xham(i,nxt)=a(i)
    3 continue
      tham=told
      call rhs(nxt)
c     get gdot from obser
      call obser(a,told,3,sdat,obs)
      dgdt=obs(islit,2)
      do 4 i=1,6
      dgdt=dgdt+h(i)*f(i,nxt)
    4 continue
c     get g from obser
      call obser(a,told,1,sdat,obs)
      g=obs(islit,2)
      if(dgdt.eq.0.00d+00) then
      write(*,101)
  101 format(2x,'PROGRAM FAILED in tstate when dgdt went to 0')
```

```
      stop
      endif
      tnew=told-g/dgdt
      tdif=dabs(tnew-told)
c     write(*,110) iter,tol,tnew,told,tdif
c 110 format(2x,'debug',/,5x,i5,e20.13,/,3(2x,e20.13))
      told=tnew
      if(iter.gt.maxit.and.tol.ge.1.00d-04) then
      write(*,100)
  100 format(2x,'PROGRAM FAILED IN TSTATE (tst couldnt converge)')
      stop
      endif
      if(iter.gt.maxit) then
      iter=0
      tol=tol*1.00d+02
      endif
      if(tdif.gt.tol) goto 2
c     if we have gotten this far  we have converged to an estimate
      tst=tnew
      do 5 i=1,6
      xst(i)=a(i)
    5 continue
      return
      end


      include'dyno2'
```

PROGRAM LISTING vecxmat

```
      subroutine vecmat(id,vin,rin,vout)
c   This routine premultiplies a square matrix by a vecttor
      double precision vin(3),rin(3,3),vout(3)
      do 10 i=1,id
      vout(i)=0.00d+00
      do 5 j=1,id
      vout(i)=vout(i)+vin(j)*rin(j,i)
    5 continue
   10 continue
      return
      end
```

APPENDIX C

Utility Program

Listings

PROGRAM LISTING   starcnv.f

```
      program starcnv
c     this program:
c        reads star data from starin
c        converts data into usable format
c        orders data in scan direction
c        creates ordered data file stardat
      double precision stara(301),stard(301),rah,ram,dras,pi
      double precision decd,decs,decm,dmin
      dimension iseq(3,301),bright(301)
      open(1,file='starin')
      open(2,file='stardat')
c     open(3,file='starold')
c     open(4,file='starmid')
      pi=3.141592654d+00
c     data reading loop
      do 1 i=1,301
      read(1,100) id,irah,iram,ras,idecd,idecm,idecs,bmag
  100 format(12x,i4,1x,i2,1x,i2,f5.1,1x,i3,1x,i2,1x,i2,1x,f5.2)
c     convert right ascension components to radians
      rah=irah
      ram=iram
      dras=ras
      stara(i)=(rah+ram/6.00d+01+dras/(6.00d+01*6.00d+01))
     +         *2.00d+00*pi/2.40d+01
c     convert declination components to radians
      decd=idecd
      decd=dabs(decd)
      decm=idecm
      decs=idecs
      stard(i)=(decd+decm/6.00d+01+decs/(6.00d+01*6.00d+01))
     +         *pi/1.80d+02
      if(idecd.lt.0) stard(i)=-stard(i)
c     here  two stars need special attention (id's 1765 1852)
      if(id.eq.1765) stard(i)=-stard(i)
      if(id.eq.1852) stard(i)=-stard(i)
      iseq(1,i)=1
      iseq(2,i)=id
      iseq(3,i)=i
      bright(i)=bmag
c     write(3,200) iseq(1,i),stara(i),stard(i),bright(i),iseq(2,i),
c    +      iseq(3,i)
    1 continue
      close(1)
c     now need to order the data
c     stard goes from -pi to pi
c     to simplify ordering process will cause stard to go from 0 to 4
      do 10 i=1,301
      if(stara(i).le.pi.and.stard(i).gt.0.00d+00)
     + stard(i)=4.00-stard(i)/pi
c  this takes data in quadrant 4 and assigns it values between 3 & 4
```

```
      if(stara(i).le.pi.and.stard(i).le.0.00d+00)
     +  stard(i)=-stard(i)/pi
c  this takes data in quadrant 1 and assigns it to between 0 and 1
      if(stara(i).gt.pi) stard(i)=stard(i)/pi+2.00d+00
c  this assigns data in quadrants 2 and 3 to values between 1 and 3
c     write(4,200) iseq(1,i),stara(i),stard(i),bright(i),iseq(2,i),
c    +   iseq(3,i)
   10 continue
c     now data has stard assigned in increasing order
c     will sort data according to stard
      do 25 l=1,301
      jmin=1
      dmin=stard(1)
   15 imin=1
      do 20 j=1,301
      if(stard(j).lt.dmin.and.stard(j).lt.5.00d+00) then
        dmin=stard(j)
        imin=j
      endif
   20 continue
      iseq(1,imin)=jmin
      stard(imin)=stard(imin)+5.00d+00
   25 continue
c     now all data has been given a sequence number
c     need to convert stard back to valid radian value and
c     print sequentially ordered data into stardat
      do 30 i=1,301
      stard(i)=stard(i)-5.00d+00
      if(stara(i).le.pi.and.stard(i).lt.1.00d+00) stard(i)=-stard(i)*pi
      if(stara(i).le.pi.and.stard(i).ge.3.00d+00)
     +  stard(i)=(4.00d+00-stard(i))*pi
      if(stara(i).gt.pi) stard(i)=(stard(i)-2.00d+00)*pi
   30 continue
      do 40 i=1,301
      do 35 k=1,301
      if(iseq(1,k).eq.i)
     +write(2,200) iseq(1,k),stara(k),stard(k),bright(k),iseq(2,k),
     +     iseq(3,k)
  200 format(1x,i3,2(2x,e20.13),2x,f5.2,2x,i4,1x,i3)
   35 continue
   40 continue
c     now write flag at end of stardat and close file
      iseq(1,1)=999
      stara(1)=0.00d+00
      stard(1)=0.00d+00
      bright(1)=0.00
      iseq(2,1)=0
      iseq(3,1)=0
      write(2,200)iseq(1,1),stara(1),stard(1),bright(1),iseq(2,1),
     +   iseq(3,1)
      close(2)
      end
```

APPENDIX D

Temporary

Data Files

```
                    TEMPORARY FILE    sitings

Slit 1 of star  77(id-2261) crossed at t= 0.9616549653323e+00
     RA =  0.1617505701057e+01       DEC= -0.1304604527215e+01

Slit 2 of star  77(id-2261) crossed at t= 0.9742219665100e+00
     RA =  0.1617505701057e+01       DEC= -0.1304604527215e+01

Slit 1 of star 130(id-6746) crossed at t= 0.3453226588421e+01
     RA =  0.4733383837519e+01       DEC= -0.5310212731254e+00

Slit 2 of star 130(id-6746) crossed at t= 0.3468579114165e+01
     RA =  0.4733383837519e+01       DEC= -0.5310212731254e+00

Slit 1 of star 133(id-6742) crossed at t= 0.3481622288535e+01
     RA =  0.4729973173213e+01       DEC= -0.5162974816282e+00

Slit 2 of star 133(id-6742) crossed at t= 0.3498661445198e+01
     RA =  0.4729973173213e+01       DEC= -0.5162974816282e+00

Slit 1 of star 160(id-6752) crossed at t= 0.4552968859622e+01
     RA =  0.4732772972169e+01       DEC=  0.4367201640005e-01

Slit 2 of star 160(id-6752) crossed at t= 0.4572944599480e+01
     RA =  0.4732772972169e+01       DEC=  0.4367201640005e-01

Slit 1 of star 168(id-6771) crossed at t= 0.4788954828677e+01
     RA =  0.4741252363620e+01       DEC=  0.1668680209229e+00

Slit 2 of star 168(id-6771) crossed at t= 0.4805517806391e+01
     RA =  0.4741252363620e+01       DEC=  0.1668680209229e+00

Slit 1 of star 177(id-6787) crossed at t= 0.5165111553339e+01
     RA =  0.4747710081728e+01       DEC=  0.3632272580715e+00

Slit 2 of star 177(id-6787) crossed at t= 0.5179998187776e+01
     RA =  0.4747710081728e+01       DEC=  0.3632272580715e+00

Slit 1 of star 219(id-6927) crossed at t= 0.6898057723155e+01
     RA =  0.4805487752251e+01       DEC=  0.1269314939363e+01

Slit 2 of star 219(id-6927) crossed at t= 0.6911541012503e+01
     RA =  0.4805487752251e+01       DEC=  0.1269314939363e+01

Slit 1 of star 232(id- 424) crossed at t= 0.7484197884076e+01
     RA =  0.5930192465628e+00       DEC=  0.1556731882109e+01

Slit 2 of star 232(id- 424) crossed at t= 0.7502045042247e+01
     RA =  0.5930192465628e+00       DEC=  0.1556731882109e+01

Slit 1 of star 243(id-2165) crossed at t= 0.8273669495085e+01
     RA =  0.1620065517349e+01       DEC=  0.1147083714065e+01
```

```
Slit 2 of star 243(id-2165) crossed at t= 0.8301688652403e+01
    RA =  0.1620065517349e+01      DEC=  0.1147083714065e+01

Slit 1 of star 248(id-2077) crossed at t= 0.8658927419005e+01
    RA =  0.1563160511522e+01      DEC=  0.9474471384320e+00

Slit 2 of star 248(id-2077) crossed at t= 0.8671526066400e+01
    RA =  0.1563160511522e+01      DEC=  0.9474471384320e+00
```

```
                  TEMPORARY FILE    stars

1          77     2261                0.9623484884174e+00
     0.1617505701057e+01            -0.1304604527215e+01

2          77     2261                0.9767395951997e+00
     0.1617505701057e+01            -0.1304604527215e+01

1          86     7228                0.1490496785873e+01
     0.5472610770051e+01            -0.1553658163371e+01

2          86     7228                0.1495759603671e+01
     0.5472610770051e+01            -0.1553658163371e+01

1         130     6746                0.3453947896009e+01
     0.4733383837519e+01            -0.5310212731254e+00

2         130     6746                0.3470336850041e+01
     0.4733383837519e+01            -0.5310212731254e+00

1         133     6742                0.3482325372939e+01
     0.4729973173213e+01            -0.5162974816282e+00

2         133     6742                0.3500374983903e+01
     0.4729973173213e+01            -0.5162974816282e+00

1         160     6752                0.4553633795184e+01
     0.4732772972169e+01             0.4367201640005e-01

2         160     6752                0.4573611172263e+01
     0.4732772972169e+01             0.4367201640005e-01

1         168     6771                0.4789667475850e+01
     0.4741252363620e+01             0.1668680209229e+00

2         168     6771                0.4806009442064e+01
     0.4741252363620e+01             0.1668680209229e+00

1         177     6787                0.5165846987642e+01
     0.4747710081728e+01             0.3632272580715e+00

2         177     6787                0.5180163123040e+01
     0.4747710081728e+01             0.3632272580715e+00

1         219     6927                0.6898785943618e+01
     0.4805487752251e+01             0.1269314939363e+01

2         219     6927                0.6910557671146e+01
     0.4805487752251e+01             0.1269314939363e+01

1         232      424                0.7484906715838e+01
     0.5930192465628e+00             0.1556731882109e+01
```

```
2          232      424                      0.7500933555433e+01
      0.5930192465628e+00               0.1556731882109e+01

1          243     2165                      0.8274430760216e+01
      0.1620065517349e+01               0.1147083714065e+01

2          243     2165                      0.8300751204491e+01
      0.1620065517349e+01               0.1147083714065e+01

1          248     2077                      0.8659603669153e+01
      0.1563160511522e+01               0.9474471364320e+00

2          248     2077                      0.8670657472455e+01
      0.1563160511522e+01               0.9474471364320e+00

1          255     2088                      0.8971052313539e+01
      0.1563778648965e+01               0.7844721693690e+00

2          255     2088                      0.8979626297276e+01
      0.1563778648965e+01               0.7844721693690e+00

1          270     2134                      0.9693320450313e+01
      0.1584664422336e+01               0.4060556986663e+00

2          270     2134                      0.9706327592432e+01
      0.1584664422336e+01               0.4060556986663e+00

1          283     2159                      0.9976422863910e+01
      0.1599972414388e+01               0.2578045231005e+00

2          283     2159                      0.9995905090170e+01
      0.1599972414388e+01               0.2578045231005e+00

0          999        0                      0.              e+00
      0.              e+00              0.              e+00
```

```
                    TEMPORARY FILE    statfile

  Slit 1 of star   77(id-2261) crossed at true time : 0.9623484884174e+0
     star data (RA,Dec):   0.1617505701057e+01  -0.1304604527215e+01
  ESTIMATION   TIME
     0.9623493674646e+00

  Slit 2 of star   77(id-2261) crossed at true time : 0.9767395951997e+0
     star data (RA,Dec):   0.1617505701057e+01  -0.1304604527215e+01
  ESTIMATION   TIME
     0.9767351245182e+00

  Slit 1 of star   86(id-7228) crossed at true time : 0.1490496785873e+0
     star data (RA,Dec):   0.5472610770051e+01  -0.1553658163371e+01
  ESTIMATION   TIME
     0.1490493158008e+01

  Slit 2 of star   88(id-7228) crossed at true time : 0.1495759603671e+0
     star data (RA,Dec):   0.5472610770051e+01  -0.1553658163371e+01
  ESTIMATION   TIME
     0.1495755293484e+01

  Slit 1 of star  130(id-6746) crossed at true time : 0.3453947896009e+0
     star data (RA,Dec):   0.4733383837519e+01  -0.5310212731254e+00
  ESTIMATION   TIME
     0.3453953314138e+01

  Slit 2 of star  130(id-6746) crossed at true time : 0.3470336850041e+0
     star data (RA,Dec):   0.4733383837519e+01  -0.5310212731254e+00
  ESTIMATION   TIME
     0.3470356589056e+01

  Slit 1 of star  133(id-6742) crossed at true time : 0.3482325372939e+0
     star data (RA,Dec):   0.4729973173213e+01  -0.5162974816282e+00
  ESTIMATION   TIME
     0.3482332418755e+01

  Slit 2 of star  133(id-6742) crossed at true time : 0.3500374983903e+0
     star data (RA,Dec):   0.4729973173213e+01  -0.5162974816282e+00
  ESTIMATION   TIME
     0.3500363708262e+01

  Slit 1 of star  160(id-6752) crossed at true time : 0.4553633795184e+0
     star data (RA,Dec):   0.4732772972169e+01   0.4367201640005e-01
  ESTIMATION   TIME
     0.4553626986230e+01

  Slit 2 of star  160(id-6752) crossed at true time : 0.4573611172263e+0
     star data (RA,Dec):   0.4732772972169e+01   0.4367201640005e-01
  ESTIMATION   TIME
     0.4573609821861e+01
```

```
Slit 1 of star  168(id-6771) crossed at true time : 0.4789667475850e+0
   star data (RA,Dec):   0.4741252363620e+01   0.1668680209229e+00
ESTIMATION   TIME
    0.4789669653552e+01

Slit 2 of star  168(id-6771) crossed at true time : 0.4806009442064e+0
   star data (RA,Dec):   0.4741252363620e+01   0.1668680209229e+00
ESTIMATION   TIME
    0.4805987380158e+01

Slit 1 of star  177(id-6787) crossed at true time : 0.5165846987642e+0
   star data (RA,Dec):   0.4747710081728e+01   0.3632272580715e+00
ESTIMATION   TIME
    0.5165861588250e+01

Slit 2 of star  177(id-6787) crossed at true time : 0.5180163123040e+0
   star data (RA,Dec):   0.4747710081728e+01   0.3632272580715e+00
ESTIMATION   TIME
    0.5180175489028e+01

Slit 1 of star  219(id-6927) crossed at true time : 0.6898785943618e+0
   star data (RA,Dec):   0.4805487752251e+01   0.1269314939363e+01
ESTIMATION   TIME
    0.6898787988382e+01

Slit 2 of star  219(id-6927) crossed at true time : 0.6910557671146e+0
   star data (RA,Dec):   0.4805487752251e+01   0.1269314939363e+01
ESTIMATION   TIME
    0.6910560210426e+01

Slit 1 of star  232(id- 424) crossed at true time : 0.7484906715838e+0
   star data (RA,Dec):   0.5930192465628e+00   0.1556731882109e+01
ESTIMATION   TIME
    0.7484890136374e+01

Slit 2 of star  232(id- 424) crossed at true time : 0.7500933555433e+0
   star data (RA,Dec):   0.5930192465628e+00   0.1556731882109e+01
ESTIMATION   TIME
    0.7500912603769e+01

Slit 1 of star  243(id-2165) crossed at true time : 0.8274430760216e+0
   star data (RA,Dec):   0.1620065517349e+01   0.1147083714065e+01
ESTIMATION   TIME
    0.8274426586969e+01

Slit 2 of star  243(id-2165) crossed at true time : 0.8300751204491e+0
   star data (RA,Dec):   0.1620065517349e+01   0.1147083714065e+01
ESTIMATION   TIME
    0.8300746570214e+01

Slit 1 of star  248(id-2077) crossed at true time : 0.8659603869153e+0
   star data (RA,Dec):   0.1563160511522e+01   0.9474471364320e+00
```

```
ESTIMATION   TIME
    0.8659615609667e+01

Slit 2 of star  248(id-2077) crossed at true time : 0.8670657472455e+0
   star data (RA,Dec):    0.1563160511522e+01   0.9474471364320e+00
ESTIMATION   TIME
    0.8670668897235e+01

Slit 1 of star  255(id-2088) crossed at true time : 0.8971052313539e+0
   star data (RA,Dec):    0.1563778648965e+01   0.7844721693690e+00
ESTIMATION   TIME
    0.8971071742029e+01

Slit 2 of star  255(id-2088) crossed at true time : 0.8979628297276e+0
   star data (RA,Dec):    0.1563778648965e+01   0.7844721693690e+00
ESTIMATION   TIME
    0.8979646825753e+01

Slit 1 of star  270(id-2134) crossed at true time : 0.9693320450313e+0
   star data (RA,Dec):    0.1584664422336e+01   0.4060556986663e+00
ESTIMATION   TIME
    0.9693318591079e+01

Slit 2 of star  270(id-2134) crossed at true time : 0.9706327592432e+0
   star data (RA,Dec):    0.1584664422336e+01   0.4060556986663e+00
ESTIMATION   TIME
    0.9706322562576e+01

Slit 1 of star  283(id-2159) crossed at true time : 0.9976422863910e+0
   star data (RA,Dec):    0.1599972414388e+01   0.2578045231005e+00
ESTIMATION   TIME
    0.9976412331844e+01

Slit 2 of star  283(id-2159) crossed at true time : 0.9995905090170e+0
   star data (RA,Dec):    0.1599972414388e+01   0.2578045231005e+00
ESTIMATION   TIME
    0.9995895530761e+01

Slit 0 of star    0(id-    0) crossed at true time : 0.            e+0
```

TEMPORARY FILE   dtfile

```
Slit 1    0.9623484884174e+00   0.9623493674646e+00
 0.4864442720058e-01  -0.1932205159943e-01    0.1842105428988e+01
 0.8109018988367e-01  -0.5827933647731e-01    0.1911392043842e+01
Slit 2    0.9767395951997e+00   0.9767351245182e+00
 0.4569618340783e-01  -0.5121895355619e+00    0.1903776241063e+01
 0.5910863206373e+00  -0.1973694216779e+00    0.1940572941823e+01
Slit 1    0.1490496785873e+01   0.1490493158008e+01
 0.1557492246503e+00  -0.2766672914790e-01    0.2847311453673e+01
 0.9201885575445e-01   0.2424185593488e-01    0.1907344165192e+01
Slit 2    0.1495759603671e+01   0.1495755293484e+01
 0.1573558862420e+00  -0.7392385874678e+00    0.2906219073380e+01
 0.6208710571477e+00   0.3098121074822e-01    0.1929486439507e+01
Slit 1    0.3453947896009e+01   0.3453953314138e+01
 0.3798249634843e+00   0.1349311647171e+00    0.6583108850559e+01
-0.4560752383671e-01   0.6487730619569e-01    0.1906631397698e+01
Slit 2    0.3470336850041e+01   0.3470356589056e+01
 0.3777336589896e+00  -0.8260423090656e+00    0.6680762625660e+01
 0.2159407732724e+00   0.5126095855667e+00    0.1893814750742e+01
Slit 1    0.3482325372939e+01   0.3482332418755e+01
 0.3712872541495e+00   0.1393191112686e+00    0.6637783625328e+01
-0.4280620224801e-01   0.6053855757213e-01    0.1906580830176e+01
Slit 2    0.3500374983903e+01   0.3500363708262e+01
 0.3682573579863e+00  -0.8167263368482e+00    0.6738283320765e+01
 0.2118472874077e+00   0.5116314532644e+00    0.1893246670447e+01
Slit 1    0.4553633795184e+01   0.4553626986230e+01
 0.4098308389756e+00   0.3014775059426e+00    0.8676865097808e+01
-0.8190872135005e-01  -0.2396945415445e-01    0.1911244868251e+01
Slit 2    0.4573611172263e+01   0.4573609821861e+01
 0.4049838617559e+00  -0.3629724168605e+00    0.8761189830986e+01
-0.1041411707509e+00   0.4857225785489e+00    0.1882859397685e+01
Slit 1    0.4789667475850e+01   0.4789669653552e+01
 0.4253882294814e+00   0.3296067895979e+00    0.9126175587661e+01
-0.9142098152716e-01  -0.4733420035519e-01    0.1913182384894e+01
Slit 2    0.4806009442064e+01   0.4805987380158e+01
 0.4211287762175e+00  -0.2434168931074e+00    0.9197154369800e+01
-0.1754135069165e+00   0.4549545354879e+00    0.1882601787411e+01
Slit 1    0.5165846987642e+01   0.5165861588250e+01
 0.4222631338391e+00   0.3640502113571e+00    0.9844920187381e+01
-0.8080188807740e-01  -0.8213376281860e-01    0.1915120155626e+01
Slit 2    0.5180163123040e+01   0.5180175489028e+01
 0.4178252646043e+00  -0.4454412348864e-01    0.99[illegible]38561799e+01
-0.2600290197587e+00   0.3923034879111e+00    0.1882016077163e+01
Slit 1    0.6898785943618e+01   0.6898787988382e+01
 0.4171570199318e+00   0.3773414278627e+00    0.1316518757451e+02
 0.6918974309387e-01  -0.9024440145528e-01    0.1913269772024e+01
Slit 2    0.6910557671146e+01   0.6910560210426e+01
 0.4133562860615e+00   0.8085852535571e+00    0.1315835468950e+02
-0.4105338359547e+00   0.5862147950630e-01    0.1884909229074e+01
Slit 1    0.7484906715838e+01   0.7484890136374e+01
 0.4941750995559e+00   0.3673521424956e+00    0.1428366558302e+02
 0.9247629102725e-01  -0.2120095753384e-01   0.1909299984220e+01
Slit 2   0.7500933555433e+01   0.7500912603769e+01
 0.4905082724879e+00   0.1036372436680e+01   0.1426862063376e+02
-0.4091182494976e+00  -0.2499335031801e-01   0.1888148661238e+01
Slit 1   0.8274430760216e+01   0.8274426586969e+01
 0.6807923593458e+00   0.3893240869105e+00   0.1578216187409e+02
 0.4871889204542e-01   0.8374251500526e-01   0.1905350700188e+01
Slit 2   0.8300751204491e+01   0.8300746570214e+01
 0.6801810618014e+00   0.1280417483592e+01   0.1577136124614e+02
-0.4110734296238e+00  -0.1236691902249e+00   0.1897009081886e+01
Slit 1   0.8659603669153e+01   0.8659615609667e+01
 0.6340809820082e+00   0.4206377147655e+00   0.1652139417257e+02
 0.4448283254201e-01   0.5222682340925e-01   0.1905470801778e+01
Slit 2   0.8670657472455e+01   0.8670668897235e+01
 0.6278340797341e+00   0.1363046033837e+01   0.1647820132617e+02
-0.3647939074809e+00  -0.2419004929217e+00   0.1903716532341e+01
Slit 1   0.8971052313539e+01   0.8971071742029e+01
 0.6592989841324e+00   0.4566537084906e+00   0.1711400335296e+02
 0.2404551269667e-01   0.5342927765907e-01   0.1905225519032e+01
Slit 2   0.8979626297276e+01   0.8979646825753e+01
 0.6518468696485e+00   0.1415990548903e+01   0.1706498503708e+02
-0.3322309710950e+00  -0.3037162451191e+00   0.1909024279934e+01
Slit 1   0.9693320450313e+01   0.9693318591079e+01
 0.7700535968677e+00   0.5654190819963e+00   0.1848285066911e+02
-0.4986784963034e-01   0.4579527009032e-01   0.1907041262433e+01
Slit 2   0.9706327592432e+01   0.9706322562576e+01
 0.7655157313732e+00   0.1470431627914e+01   0.1844586270636e+02
-0.2492946726603e+00  -0.4214947617480e+00   0.1923497212728e+01
Slit 1   0.9976422863910e+01   0.9976412331844e+01
 0.8304921808119e+00   0.6116498571258e+00   0.1901795597927e+02
-0.9055515367955e-01   0.3275843280650e-01   0.1909270651232e+01
Slit 2   0.9995905090170e+01   0.9995895530761e+01
 0.8297862796920e+00   0.1459788218009e+01   0.1899702710652e+02
-0.2195837995208e+00  -0.4619460510249e+00   0.1930420940248e+01
     0   0.               e+00   0.9995895530761e+01
         END OF DATA
```

```
                 TEMPORARY FILE     errlist

      0.                    0.0106771
      0.                    0.0007045
      0.                    0.0002392
      0.                    0.0002442
      0.                    0.0002442
    -12.0000000             0.0002423
    -24.0000000             0.0002861
    -36.0000000             0.0003636
    -48.0000000             0.0004594
    -60.0000000             0.0005652
    -72.0000000             0.0006854
    -84.0000000             0.0008130
    -96.0000000             0.0009475
   -108.0000000             0.0010858
   -120.0000000             0.0012373
   -132.0000000             0.0013960
   -144.0000000             0.0015651
   -156.0000000             0.0017361
   -168.0000000             0.0019210
   -180.0000000             0.0021085
   -192.0000000             0.0023101
   -204.0000000             0.0025152
   -216.0000000             0.0027397
   -228.0000000             0.0029629
   -240.0000000             0.0032055
   -252.0000000             0.0034426
   -264.0000000             0.0037040
   -276.0000000             0.0039655
   -288.0000000             0.0042370
   -300.0000000             0.0045274
   -312.0000000             0.0048187
   -324.0000000             0.0051202
   -336.0000000             0.0054319
   -348.0000000             0.0057539
   -360.0000000             0.0060781
   -360.0000000             0.0060781
   -360.0000000             0.0001575
   -360.0000000             0.0002345
   -360.0000000             0.0002346
   -372.0000000             0.0002410
   -384.0000000             0.0002627
   -396.0000000             0.0002975
   -408.0000000             0.0003522
   -420.0000000             0.0004095
   -432.0000000             0.0004858
   -444.0000000             0.0005527
   -456.0000000             0.0006387
   -468.0000000             0.0007433
   -480.0000000             0.0008485
   -492.0000000             0.0009723
```

-504.0000000	0.0011057
-516.0000000	0.0012327
-528.0000000	0.0013708
-540.0000000	0.0015345
-552.0000000	0.0016942
-564.0000000	0.0018783
-576.0000000	0.0020597
-588.0000000	0.0022644
-600.0000000	0.0024559
-612.0000000	0.0026816
-624.0000000	0.0029068
-636.0000000	0.0031430
-648.0000000	0.0033902
-660.0000000	0.0036484
-672.0000000	0.0039174
-684.0000000	0.0041973
-696.0000000	0.0044799
-708.0000000	0.0047817
-720.0000000	0.0050941
-720.0000000	0.0050941
-720.0000000	0.0002543
-720.0000000	0.0003894
-720.0000000	0.0003895
-732.0000000	0.0003962
-744.0000000	0.0004105
-756.0000000	0.0004280
-768.0000000	0.0004540
-780.0000000	0.0004855
-792.0000000	0.0005267
-804.0000000	0.0005784
-816.0000000	0.0006392
-828.0000000	0.0007120
-840.0000000	0.0007962
-852.0000000	0.0008926
-864.0000000	0.0010000
-876.0000000	0.0011190
-888.0000000	0.0012506
-900.0000000	0.0013923
-912.0000000	0.0015452
-924.0000000	0.0017107
-936.0000000	0.0018858
-948.0000000	0.0020755
-960.0000000	0.0022744
-972.0000000	0.0024819
-984.0000000	0.0027050
-996.0000000	0.0029337
-1008.0000000	0.0031733
-1020.0000000	0.0034285
-1032.0000000	0.0036947
-1044.0000000	0.0039719
-1056.0000000	0.0042533
-1068.0000000	0.0045516

-1080.0000000	0.0048540
-1080.0000000	0.0048540
-1080.0000000	0.0004429
-1080.0000000	0.0002896
-1080.0000000	0.0002909
-1092.0000000	0.0003007
-1104.0000000	0.0003125
-1116.0000000	0.0003270
-1128.0000000	0.0003459
-1140.0000000	0.0003710
-1152.0000000	0.0004044
-1164.0000000	0.0004480
-1176.0000000	0.0005066
-1188.0000000	0.0005698
-1200.0000000	0.0006520
-1212.0000000	0.0007432
-1224.0000000	0.0008466
-1236.0000000	0.0009679
-1248.0000000	0.0010943
-1260.0000000	0.0012325
-1272.0000000	0.0013887
-1284.0000000	0.0015488
-1296.0000000	0.0017202
-1308.0000000	0.0019094
-1320.0000000	0.0021022
-1332.0000000	0.0023059
-1344.0000000	0.0025204
-1356.0000000	0.0027519
-1368.0000000	0.0029875
-1380.0000000	0.0032338
-1392.0000000	0.0035035
-1404.0000000	0.0037702
-1416.0000000	0.0040475
-1428.0000000	0.0043354
-1440.0000000	0.0046399
-1440.0000000	0.0046399
-1440.0000000	0.0012331
-1440.0000000	0.0015946
-1440.0000000	0.0016011
-1452.0000000	0.0016302
-1464.0000000	0.0016748
-1476.0000000	0.0017475
-1488.0000000	0.0018238
-1500.0000000	0.0019357
-1512.0000000	0.0020673
-1524.0000000	0.0022167
-1536.0000000	0.0023823
-1548.0000000	0.0025281
-1560.0000000	0.0027203
-1572.0000000	0.0028891
-1584.0000000	0.0031057
-1596.0000000	0.0033726

-1608.0000000	0.0035745
-1620.0000000	0.0038259
-1632.0000000	0.0040882
-1644.0000000	0.0043222
-1656.0000000	0.0046059
-1668.0000000	0.0049000
-1680.0000000	0.0051663
-1692.0000000	0.0054815
-1704.0000000	0.0058072
-1716.0000000	0.0061433
-1728.0000000	0.0064898
-1740.0000000	0.0068098
-1752.0000000	0.0071772
-1764.0000000	0.0075192
-1776.0000000	0.0079077
-1788.0000000	0.0082717
-1800.0000000	0.0086813
-1800.0000000	0.0086813
-1800.0000000	0.0003140
-1800.0000000	0.0002814
-1800.0000000	0.0002810
-1812.0000000	0.0002731
-1824.0000000	0.0002903
-1836.0000000	0.0003249
-1848.0000000	0.0004170
-1860.0000000	0.0005298
-1872.0000000	0.0006551
-1884.0000000	0.0007381
-1896.0000000	0.0008296
-1908.0000000	0.0009787
-1920.0000000	0.0010865
-1932.0000000	0.0012035
-1944.0000000	0.0013754
-1956.0000000	0.0015098
-1968.0000000	0.0016975
-1980.0000000	0.0018083
-1992.0000000	0.0020128
-2004.0000000	0.0022254
-2016.0000000	0.0024462
-2028.0000000	0.0025990
-2040.0000000	0.0028377
-2052.0000000	0.0030141
-2064.0000000	0.0032709
-2076.0000000	0.0035364
-2088.0000000	0.0037457
-2100.0000000	0.0040298
-2112.0000000	0.0042620
-2124.0000000	0.0045648
-2136.0000000	0.0048197
-2148.0000000	0.0051412
-2160.0000000	0.0054718
86400.0000000	0.0054718

TEMPORARY FILE   errplot

```
0.       1.00000"+"
0.       0.06598"+"
0.       0.02240"+"
0.       0.02287"+"
0.       0.02287"+"
0.00556 0.02269"+"
0.01111 0.02680"+"
0.01667 0.03405"+"
0.02222 0.04303"+"
0.02778 0.05294"+"
0.03333 0.06419"+"
0.03889 0.07614"+"
0.04444 0.08874"+"
0.05000 0.10169"+"
0.05556 0.11588"+"
0.06111 0.13075"+"
0.06667 0.14658"+"
0.07222 0.16260"+"
0.07778 0.17992"+"
0.08333 0.19748"+"
0.08889 0.21636"+"
0.09444 0.23557"+"
0.10000 0.25660"+"
0.10556 0.27750"+"
0.11111 0.30022"+"
0.11667 0.32243"+"
0.12222 0.34891"+"
0.12778 0.37140"+"
0.13333 0.39683"+"
0.13889 0.42403"+"
0.14444 0.45131"+"
0.15000 0.47955"+"
0.15556 0.50874"+"
0.16111 0.53890"+"
0.16667 0.56927"+"
0.16667 0.56927"+"
0.16667 0.01475"+"
0.16667 0.02196"+"
0.16667 0.02197"+"
0.17222 0.02257"+"
0.17778 0.02460"+"
0.18333 0.02786"+"
0.18889 0.03299"+"
0.19444 0.03835"+"
0.20000 0.04550"+"
0.20556 0.05176"+"
0.21111 0.05982"+"
0.21667 0.06962"+"
0.22222 0.07947"+"
0.22778 0.09106"+"
```

```
0.23333 0.10356"+"
0.23889 0.11545"+"
0.24444 0.12839"+"
0.25000 0.14372"+"
0.25556 0.15868"+"
0.26111 0.17592"+"
0.26667 0.19291"+"
0.27222 0.21208"+"
0.27778 0.23002"+"
0.28333 0.25115"+"
0.28889 0.27225"+"
0.29444 0.29437"+"
0.30000 0.31752"+"
0.30556 0.34170"+"
0.31111 0.36690"+"
0.31667 0.39311"+"
0.32222 0.41958"+"
0.32778 0.44785"+"
0.33333 0.47711"+"
0.33333 0.47711"+"
0.33333 0.02382"+"
0.33333 0.03647"+"
0.33333 0.03648"+"
0.33889 0.03711"+"
0.34444 0.03845"+"
0.35000 0.04009"+"
0.35556 0.04252"+"
0.36111 0.04547"+"
0.36667 0.04933"+"
0.37222 0.05417"+"
0.37778 0.05987"+"
0.38333 0.06668"+"
0.38889 0.07457"+"
0.39444 0.08360"+"
0.40000 0.09366"+"
0.40556 0.10480"+"
0.41111 0.11713"+"
0.41667 0.13040"+"
0.42222 0.14472"+"
0.42778 0.16022"+"
0.43333 0.17662"+"
0.43889 0.19439"+"
0.44444 0.21302"+"
0.45000 0.23245"+"
0.45556 0.25335"+"
0.46111 0.27477"+"
0.46667 0.29721"+"
0.47222 0.32111"+"
0.47778 0.34604"+"
0.48333 0.37200"+"
0.48889 0.39836"+"
0.49444 0.42630"+"
```

```
0.50000 0.45462"+"
0.50000 0.45462"+"
0.50000 0.04148"+"
0.50000 0.02712"+"
0.50000 0.02725"+"
0.50556 0.02816"+"
0.51111 0.02927"+"
0.51667 0.03063"+"
0.52222 0.03240"+"
0.52778 0.03475"+"
0.53333 0.03788"+"
0.53889 0.04196"+"
0.54444 0.04745"+"
0.55000 0.05337"+"
0.55556 0.06107"+"
0.56111 0.06961"+"
0.56667 0.07929"+"
0.57222 0.09065"+"
0.57778 0.10249"+"
0.58333 0.11543"+"
0.58889 0.13006"+"
0.59444 0.14506"+"
0.60000 0.16111"+"
0.60556 0.17883"+"
0.61111 0.19689"+"
0.61667 0.21597"+"
0.62222 0.23606"+"
0.62778 0.25774"+"
0.63333 0.27980"+"
0.63889 0.30287"+"
0.64444 0.32813"+"
0.65000 0.35311"+"
0.65556 0.37908"+"
0.66111 0.40605"+"
0.66667 0.43457"+"
0.66667 0.43457"+"
0.66667 0.11549"+"
0.66667 0.14935"+"
0.66667 0.14996"+"
0.67222 0.15268"+"
0.67778 0.15686"+"
0.68333 0.16367"+"
0.68889 0.17081"+"
0.69444 0.18129"+"
0.70000 0.19362"+"
0.70556 0.20761"+"
0.71111 0.22312"+"
0.71667 0.23678"+"
0.72222 0.25478"+"
0.72778 0.27059"+"
0.73333 0.29087"+"
0.73889 0.31587"+"
```

```
0.74444 0.33478"+"
0.75000 0.35833"+"
0.75556 0.38289"+"
0.76111 0.40481"+"
0.76667 0.43138"+"
0.77222 0.45893"+"
0.77778 0.48387"+"
0.78333 0.51339"+"
0.78889 0.54389"+"
0.79444 0.57537"+"
0.80000 0.60782"+"
0.80556 0.63779"+"
0.81111 0.67221"+"
0.81667 0.70424"+"
0.82222 0.74062"+"
0.82778 0.77471"+"
0.83333 0.81308"+"
0.83333 0.81308"+"
0.83333 0.02941"+"
0.83333 0.02636"+"
0.83333 0.02632"+"
0.83889 0.02558"+"
0.84444 0.02719"+"
0.85000 0.03043"+"
0.85556 0.03906"+"
0.86111 0.04962"+"
0.86667 0.06136"+"
0.87222 0.06913"+"
0.87778 0.07770"+"
0.88333 0.09166"+"
0.88889 0.10176"+"
0.89444 0.11272"+"
0.90000 0.12882"+"
0.90556 0.14141"+"
0.91111 0.15899"+"
0.91667 0.16936"+"
0.92222 0.18852"+"
0.92778 0.20843"+"
0.93333 0.22911"+"
0.93889 0.24342"+"
0.94444 0.26577"+"
0.95000 0.28230"+"
0.95556 0.30635"+"
0.96111 0.33121"+"
0.96667 0.35082"+"
0.97222 0.37742"+"
0.97778 0.39917"+"
0.98333 0.42753"+"
0.98889 0.45141"+"
0.99444 0.48152"+"
1.00000 0.51248"+"
```

VITA

Jack Taylor was born in Monticello, New York, 16 October, 1953. He graduated from Goshen Central High School, New York, in 1971. After entering the Air Force in July, 1972, he performed enlisted duty for eight years, first as a flight instrument trainer technician, and later as a civil engineering site development specialist. He attended Auburn University under the Air Force Airmen Education and Commissioning Program, and was awarded a Bachelor of Science Degree in Aerospace Engineering in March, 1982. After receiving an Air Force commission in June, 1982, he served for the next three years as a satellite systems engineer in Air Force military communications. He was enrolled in the Air Force Institute of Technology Master's Degree program in June, 1985.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

A179 427

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION / DOWNGRADING SCHEDULE

3. DISTRIBUTION / AVAILABILITY OF REPORT
Approved for public release;
distribution unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
AFIT/GA/AA/86D-14

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
School of Engineering

6b. OFFICE SYMBOL *(If applicable)*
AFIT/ENY

7a. NAME OF MONITORING ORGANIZATION

6c. ADDRESS *(City, State, and ZIP Code)*
Air Force Institute of Technology
Wright-Patterson AFB, Ohio 45433

7b. ADDRESS *(City, State, and ZIP Code)*

8a. NAME OF FUNDING / SPONSORING ORGANIZATION

8b. OFFICE SYMBOL *(If applicable)*

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS *(City, State, and ZIP Code)*

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO	WORK UNIT ACCESSION NO.

11. TITLE *(Include Security Classification)*
STAR SENSOR FINE ATTITUDE POINTING MODEL FOR A
SPINNING GEOSYNCHRONOUS SATELLITE (UNCLASSIFIED)

12. PERSONAL AUTHOR(S)
John V. Taylor IV, M.S., Capt, USAF
THESIS CHAIRMAN:
Dr. William E. Wiesel

13a. TYPE OF REPORT
MS Thesis

13b. TIME COVERED
FROM ________ TO ______

14. DATE OF REPORT *(Year, Month, Day)*
1986 December

15. PAGE COUNT
150

16. SUPPLEMENTARY NOTATION

Approved for public release: IAW AFR 190-1.
Lynn Wolaver 13 Mar 87
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (AU)
Wright-Patterson AFB OH 45433

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
22	02,03	
17	07	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Spacecraft Attitude Determination, Star Sensors,
Star Trackers, Least Squares Estimation,
Autonomous Spacecraft Navigation

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

A fine attitude determination model is developed for a spinning geosynchronous satellite, based upon stellar observations from a v-slit star scanner and ground supplied ephemerides. A true attitude model is determined through numerical integration of Euler's moment equations for a torque-free, rigid, axisymmetric body, and kinematic relations which consist of pitch, roll, and yaw orientation angles. Next, the closed form solutions to the Euler equations are coupled with first-order approximate solutions of the kinematic relations to develop a second order kinematics model. Observation relations, relating stellar slit-plane crossing times,(a priori star identification assumed), to attitude states, are then developed. Finally, a nonlinear least squares estimation algorithm is used to identify the full satellite attitude state. Simulation is tested for single scan and multiple scan capability using exact and noise ridden data.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL
Dr. William E. Wiesel, Professor

22b. TELEPHONE *(Include Area Code)*
513-255-3517

22c. OFFICE SYMBOL
AFIT/ENY

UNCLASSIFIED